Invitació al Logo

Joan Josep Ordinas Rosa

1993

Invitació al Logo

Joan Josep Ordinas Rosa

E-mail: jjor@piec.xtec.es

1a edició. Barcelona 1993.



Generalitat de Catalunya
Departament d'Ensenyament
Programa d'Informàtica Educativa

PREFACI

Invitació al Logo és un curs divulgatiu sobre el llenguatge Logo i les seves aplicacions educatives. Les intencions generals del curs són:

- Oferir una descripció acurada de les característiques fonamentals del llenguatge Logo i el seu entorn de treball. El WIN-LOGO, darrera implementació del llenguatge en idioma català és la versió del Logo amb què es realitzaran les pràctiques del curs.

- Propiciar la reflexió sobre el valor pedagògic del Logo i les possibilitats de la seva inserció dins de l'actual ordenació del sistema educatiu. Conèixer la filosofia d'aprenentatge del Logo és condició necessària per valorar el sentit i objectiu de les pràctiques realitzades davant de l'ordinador.

- Mostrar diverses experiències educatives realitzades amb el Logo a les escoles catalanes. Són els exemples d'ús real del Logo a les escoles els que han de completar la visió del llenguatge que aquest curs vol oferir.

- Estimular l'ús del Logo en els diferents nivells educatius. Una característica del Logo és la de no tenir límit, ni per baix ni per dalt. Des de l'inici de l'escolarització, fins als primers anys de la universitat, el Logo pot ensenyar i ajudar a aprendre.

Invitació al Logo s'adreça a tots els educadors de Catalunya amb curiositat per conèixer les possibilitats d'aplicació de les noves tecnologies a l'escola. El Logo

representa un paradigma de primera categoria dins de l'ensenyament realitzat amb ordinadors, i qualsevol reflexió en aquest camp ha de tenir com a referència, tan al llenguatge com a les experiències amb ell realitzades.

És avui, més de dues dècades després del naixement del Logo, quan comença a generalitzar-se la presència dels ordinadors a les escoles, i es fa possible plantejar-se l'ús de la informàtica com un recurs didàctic d'abast quotidià. Desgraciadament, les expectatives inicials sobre els beneficis que l'ús del Logo havia de proporcionar no han estat satisfetes del tot al llarg d'aquest temps, potser a causa més d'un excés d'optimisme inicial que a les limitacions pròpies del llenguatge. El cert és que s'ha estès un cert estat de desànim, fins i tot abans que el Logo arribi a ser conegut de forma generalitzada en l'àmbit educatiu.

Però si analitzem els recursos de què disposem, no es troba cap raó pel desencís. Els ordinadors que podem trobar a les escoles tenen una potència fins fa pocs anys reservada als centres d'investigació més sofisticats. Avui en dia tenim una versió moderna del Logo, capaç de treure profit de les capacitats gràfiques dels ordinadors més moderns, i disposem d'àmplia informació sobre les experiències fetes els darrers anys amb aquest llenguatge.

Les condicions per experimentar amb Logo són doncs immillorables. Ara, quan ens acostem a la fi del mil·lenni, és el moment definitiu per decidir si Logo ha de seguir tenint un lloc dins l'escola. Tots els educadors que vulgueu participar en aquest procés esteu convidats, i podeu fer servir aquesta *Invitació al Logo*.

CONTINGUT

ABANS DE COMENÇAR

Per tal que pugueu aprofitar al màxim la lectura d'aquest manual, cal abans que tingueu presents algunes convencions i criteris emprats en la seva confecció.

Convencions i estil

Al llarg d'aquest document es fan servir, amb la finalitat de facilitar la lectura del text, diversos símbols, tipus de lletra i convencions tipogràfiques.

El nom de les tecles

Tots els noms de tecles estan escrits en lletra versaleta, és a dir, lletra majúscula amb la mateixa alçada de les lletres minúscules del mateix cos. Per exemple, la tecla Control apareixerà mencionada com a CTRL i la tecla Escapar com ESC. És possible que les tecles del vostre ordinador no estiguin retolades tal com s'indica en aquest manual, i de fet, podeu trobar algunes tecles que no tenen en cap cas el seu nom escrit a sobre, tal com succeeix amb les tecles RETORN o RETROCÉS.

A vegades, dues o més tecles s'han de prémer alhora. Hi ha tecles, com ara CTRL i ALT, que sempre es fan servir combinades amb altres. Per indicar que dues tecles s'han de prémer alhora, les escriurem unides amb el símbol de la suma. Així direm que les tecles CTRL+X permeten abandonar WIN-LOGO, o que prement les tecles MAJÚSCULES+RETORN s'insereix en el text un final de línia virtual.

Les tecles de direcció, un subconjunt de les tecles de desplaçament, s'anomenen fent referència a la fletxa dibuixada a la part superior de tecla. Així parlarem de tecla FLETXA AMUNT, FLETXA AVALL, FLETXA ESQUERRA i FLETXA DRETA. L'expressió "tecles de direcció" descriu aquestes quatre tecles de forma col·lectiva. La resta de tecles de desplaçament es mencionen amb el seu nom específic: INICI, FI, PÀGINA AMUNT i PÀGINA AVALL.

El text

Els noms de fitxers s'escriuran en lletres majúscules. Direm, per exemple, que WLOGO.INI és el nom del fitxer de configuració del WIN-LOGO.

La menció als diversos elements que apareixen a l'entorn de treball del WIN-LOGO, noms i comandes dels menús, botons dels quadres de diàleg, etc., es farà en lletra negreta. Explicarem, per exemple, que prémer la tecla F10 produeix el mateix efecte que seleccionar la comanda **Interpreta** del menú **Edició**.

Els fragments de llenguatge Logo s'escriuran amb el tipus `courier`. Aquest és un tipus en què totes les lletres fan la mateixa amplada. WIN-LOGO fa servir també lletra d'aquesta mena, pel que serà fàcil associar els exemples escrits en aquest manual amb la forma en què es visualitzarien a l'entorn WIN-LOGO. Així direm, per exemple, que `avança` i `gira.dreta` són sempre les dues primeres ordres de la tortuga que s'ensenyen.

En reproduir les instruccions escrites dins la finestra de Treball, veureu també escrit el símbol d'atenció del Logo, un interrogant (?). D'aquesta manera la instrucció es visualitzarà de la mateixa forma que a la mateixa pantalla

de l'ordinador. En cap cas heu d'escriure l'interrogant quan, en les vostres pràctiques, teclegeu les instruccions.

En presentar instruccions que escriguin dins la finestra de Text, reproduirem el text escrit a sota de la línia d'exemple que l'ha produït. El distingireu de les instruccions per no tenir escrit l'interrogant al principi de la línia. Aquesta és la forma en què es visualitza el text en les implementacions tradicionals del Logo.

Hi ha dos caràcters que poden aparèixer al final de les línies de codi i que no heu d'escriure directament (farem servir la paraula *codi* per referir-nos a qualsevol fragment de text escrit amb llenguatge Logo). El caràcter de continuació de línia (») és sempre escrit pel propi WIN-LOGO, per "plegar" les línies massa llargues. El caràcter de fi de línia virtual («) apareix en prémer les tecles MAJÚSCULES+RETORN, que farem servir per tallar les línies massa llargues i obtenir un sagnat clar del codi.

Símbols

Aquest manual fa servir un recurs gràfic per guiar la lectura del text. Consisteix a posar al costat d'alguns paràgrafs, al marge esquerre, uns petits símbols que identifiquen la naturalesa del text: comentari, exercici, metodologia, etc. És a dir, qualifiquen al text segons les seves característiques, classificant-lo en pocs i definits grups.

	Comentari. Temps d'aturar-se a reflexionar, plantejar algun punt de discussió o fer algun aclariment informal. És important que discutiu entre vosaltres tot allò que us agradi o disgusti, el que us costi d'entendre, les idees que Logo us suggereixi...
	Definició. Els nous conceptes necessiten una definició adequada. És un text al qual cal tornar periòdicament, per refrescar i consolidar les idees.
	Drecera. No sempre l'ordre seqüencial de lectura és el més adequat. Per exemple, hi ha textos que han de figurar dins del manual, vàlids per a usuaris avançats o com a material de referència, però que es poden deixar de banda en una primera lectura. El símbol de drecera us indica que podeu desviar-vos del camí lineal a què forcen les pàgines del text.
	Exercici. És moment, metafòricament parlant, de fer punta al llapis per començar a fer els deures. Tot el manual està esquitxat d'exercicis que heu de realitzar, alguns a l'aula, i altres en les vostres hores de pràctiques. No us en deixeu cap per fer!

	**Metodologia.** Per aprendre i aplicar adequadament un llenguatge de programació com el Logo fan falta orientacions i consells. Cal trobar la direcció correcte, i el mètode de treball adequat. Tota orientació metodològica present al text anirà senyalada amb aquest símbol.

A les properes pàgines, trobareu altres símbols a més dels recollits a la llista anterior. Tots aquests altres símbols tenen un significat obvi, i es fan servir de forma menys formal i rigorosa que els comentats anteriorment, únicament en qualitat de comentari gràfic

Usualment, els símbols afecten al paràgraf que acompanyen, encara que a vegades poden estendre's als paràgrafs següents. Si un símbol acompanya al primer paràgraf d'una secció, el seu significat s'aplica a tot el text de la mateixa.

La presència d'aquests símbols no s'ha de viure com una complicació afegida. Els podeu ignorar per complet, ja que en cap moment són necessaris per seguir el text. Ara bé, una vegada us acostumeu a ells, us ajudaran a treure el màxim profit del manual.

Abreviatures



Moltes primitives tenen dos noms, un de complet i l'abreujat. Les abreviatures existeixen dins del Logo per facilitar el treball en el mode immediat, quan escrivim i executem les ordres una per una, i eviten haver d'escriure llargues paraules. El que és un avantatge en el mode immediat és un inconvenient a l'hora d'escriure els programes dins de l'editor: la lectura de les abreviatures és força difícil, i el seu ús dona lloc a programes absolutament incomprensibles.

Quan en aquest manual es presenti el nom d'una nova primitiva, si disposa d'abreviatura, també serà mencionada. Després, però, no s'hi tornarà a fer referència en cap cas. La claredat en el codi ha de ser una fita en el moment d'escriure en qualsevol llenguatge de programació. Per intentar assolir aquest objectiu, no es faran servir abreviatures en els exemples d'aquest manual.

Execució del WIN-LOGO

Encara que basat en el WIN-LOGO, *Invitació al Logo* no incidirà massa sobre els aspectes més particulars d'aquesta implementació del llenguatge. En especial, no es tractaran els detalls que ja estan recollits a la documentació del WIN-LOGO, ni tampoc es farà esment de les múltiples variants que pot tenir la instal·lació del programa en el disc dur de l'ordinador.

Suposarem que el WIN-LOGO està instal·lat a l'ordinador amb què heu de treballar. Si no és així, i heu de resoldre aquest problema personalment, haureu de consultar la documentació original del WIN-LOGO per saber com

instal·lar el programa. Trobareu informació suficient al segon capítol del *Manual de l'Usuari*.

La forma d'executar el WIN-LOGO pot ser diversa, i depèn de com estigui configurat l'ordinador amb què treballeu. Si sou vosaltres els que heu instal·lat el programa també sabreu com executar-lo. Si no és així, demaneu ajuda a l'encarregat del manteniment dels ordinadors o als vostres companys per solucionar-ho.

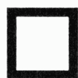
Interacció amb el programa

L'entorn de treball del WIN-LOGO es compon de diverses finestres, cada una amb una funció i comportament particulars, i de diversos menús desplegables agrupats a la barra de menús.

Encara que en un moment donat hi hagin diverses finestres visibles, només una d'elles és la finestra activa. La reconeixereu per tenir, en color contrastat, la línia on està escrit el seu nom. Per activar una finestra, només l'heu de clicar amb el ratolí.

La barra de menús es troba a la segona línia de la pantalla, i conté els noms dels menús disponibles en cada moment.

Finestres



Les tres finestres que trobeu obertes quan el programa arrenca, si el fitxer WLOGO.INI original no ha estat modificat en aquest aspecte, són les finestres de Treball, Gràfics i Text. Amb el ratolí podeu modificar la mida i posició de les finestres, i canviar la finestra activa, que inicialment és la de Treball.

La finestra de Treball és on escriureu les instruccions que vulgueu fer avaluar per l'intèrpret una a una. Molt freqüentment, aquestes instruccions produeixen algun efecte dins les finestres de Gràfics o Text.

La finestra de Gràfics conté el món de la tortuga. Les ordres de la tortuga, que aviat aprendreu, dibuixen dins d'aquesta finestra.

La finestra de Text mostra els missatges que el WIN-LOGO pot produir, i també és útil per escriure-hi text o per llegir-ne.

També cal mencionar la finestra d'Edició. Encara que no visible inicialment, hi treballareu molt sovint, quan escriviu els vostres programes.

Barra de menús



Els menús desplegables ofereixen un mètode visual de fer seleccions d'entre un conjunt determinat de comandes.

La barra de menús es troba a la segona línia de la pantalla. Hi ha escrits els noms dels menús disponibles en cada moment, en funció de la finestra que estigui activada. Tots els noms tenen una lletra destacada, amb un color diferent de les altres. Per obrir un menú, heu de clicar amb el ratolí damunt del seu nom o prémer la tecla ALT en combinació de la lletra destacada. Així, per exemple, en prémer les tecles ALT+A s'obre el menú **Ajuda**.

Ratolí



Aquest dispositiu de nom tan curiós facilita molt la feina dins del l'entorn de treball del WIN-LOGO. Per exemple, activar una finestra és una operació tan simple com clicar una vegada al damunt seu. També podem, fent servir el ratolí, obrir qualsevol menú i triar-ne una comanda.

El ratolí controla un punter a la pantalla. Aquest punter es mou en desplaçar el ratolí per sobre d'una superfície plana. Les tècniques fonamentals per treballar amb el ratolí són les següents:

Desplaçar. És l'acció de moure el ratolí per sobre de la taula, sense aixecar-lo.

Assenyalar. És l'acció de desplaçar el ratolí, per situar el punter a sobre d'un element de la pantalla.

Prémer. És l'acció de pressionar els botons del ratolí.

Clicar. És l'acció combinada d'assenyalar un element a la pantalla, i després prémer i alliberar ràpidament el botó del ratolí.

Arrossegar. És l'acció de mantenir premut el botó del ratolí, mentre desplacem el ratolí.

Teclat



El teclat és el principal medi de què disposem per comunicar-nos amb el Logo. Amb ell teclegem les instruccions, seleccionem comandes dels menús, escrivim programes o text dins la finestra d'edició, etc. Tindreu temps de familiaritzar-vos amb el teclat al mateix temps que l'utilitzeu. De tota manera, abans de començar cal que conegueu el funcionament d'algunes tecles importants.

- RETORN. Aquesta tecla te funcions diferents segons el context. En general podríem dir que és la tecla executiva. Dins la finestra de Treball acaba les instruccions; dins d'un quadre de diàleg equival a triar l'opció d'acceptació; dins de la finestra d'Edició permet inserir una nova línia, etc.
- ESC. Permet aturar l'execució dels programes, retornant-nos el control i fent activa la finestra de Treball. Dins d'un quadre de diàleg equival a triar l'opció de cancel·lació.

Menús

Les tecles que heu de conèixer per fer servir eficaçment els menús són les següents:

- Per obrir un menú s'ha de prémer la tecla ALT, combinada amb la lletra que apareix destacada al nom del menú.
- Les tecles FLETXA DRETA i FLETXA ESQUERRA tanquen el menú obert i obren el del seu costat.
- Les tecles INICI, FI, FLETXA AMUNT i FLETXA AVALL canvien la comanda seleccionada dins del menú obert.
- La tecla RETORN fa executar la comanda seleccionada.
- La tecla ESC tanca el menú sense executar cap comanda.

Com treballar

Ara que us heu d'enfrontar a un període d'aprenentatge, d'una disciplina potser totalment nova per vosaltres, pot

ser molt convenient reconsiderar les tècniques a aplicar per assolir el màxim profit dels vostres estudis.

El manual



Aquestes pàgines que esteu llegint accepten diverses aproximacions. Podeu simplement llegir-les i, més tard practicar amb l'ordinador, o llegir-les alhora que practiqueu els seu contingut. Potser, el millor serà que feu les dues coses...

És molt important que torneu a llegir, diverses vegades, aquest manual. Aspectes als que no heu donat importància en una primera lectura, poden ser fonamentals en un moment diferent del vostre aprenentatge

Per altra banda, no heu de deixar de consultar la documentació original del WIN-LOGO, ni de llegir altres llibres sobre el Logo, l'ús de la informàtica a l'escola i la utilització i programació dels ordinadors.

L'ajuda



El WIN-LOGO disposa d'un complet sistema d'ajuda. Per acedir-hi podeu prémer la tecla F1 o obrir el menú **Ajuda**. Les tres comandes d'aquest menú, **Índex**, **Busca referències...** i **Informa primitiva** ofereixen diferents funcionalitats. Podeu accedir a l'índex principal de l'ajuda, i des d'aquest punt navegar per diverses pantalles informatives; podeu cercar tots els temes relacionats amb una primitiva o un tema donats, i podeu saltar directament a la pantalla d'ajuda de la primitiva sobre la qual es troba el cursor.

Familiaritzeu-vos amb l'ajuda del WIN-LOGO des del primer dia. No deixeu de consultar-la, i abans de preguntar als vostres companys o al professor del curs, "pregunteu" al menú **Ajuda**.

Exercicis



No insistirem prou sobre la conveniència de fer amb regularitat els exercicis continguts en aquest manual. El Logo és molt fàcil si practiqueu sovint, però si no ho feu us pot semblar un galimaties sense sentit. Els paràgrafs següents ofereixen algunes reflexions que us poden ajudar a avançar en el vostre coneixement del llenguatge.

Realitzeu els exercicis directament davant de l'ordinador, però si cal agafeu paper i llapis per enfrontar-vos a problemes especialment complexos per vosaltres. Feu diagrames, escriviu parts del programa, i torneu a l'ordinador amb les idees més clares.

A vegades trobareu que, per resoldre un problema li heu de donar voltes uns quants dies. Això és normal. No abandoneu els exercicis si no els podeu resoldre en una sola sessió de treball. Torneu-hi.

Plantegeu-vos problemes del vostre gust. Quan arribeu a posar-vos exercicis a la mida del vostre nivell d'aprenentatge, res us impedirà que acabeu dominant el Logo.

Feu contínuament intercanvi de dubtes i solucions amb els altres assistents al curs. Intercanvieu els vostres programes. Compartiu la vostra experiència i el vostre treball. Tothom hi sortirà guanyant.

Disquets



El vostre treball s'ha de desar en fitxers, i aquests en el disc dur de l'ordinador o en disquets. Hi ha diverses raons perquè trieu aquesta darrera possibilitat. La més important és que no sempre treballareu amb el mateix ordinador, però la vostra feina us haurà d'acompanyar. A més, encara que treballeu sempre amb la mateixa màquina, desar els fitxers simultàniament al disc dur i al disquet us ofereix una doble seguretat (recordeu que la fiabilitat dels suports magnètics d'informació és molt petita).

Si no sabeu com podeu formatar els disquets o com duplicar-los, o no coneixeu les ordres del sistema operatiu que permeten copiar fitxers, navegar entre els diferents subdirectoris i veure'n el contingut, consulteu la documentació adequada o demaneu ajuda als vostres companys.

1

LA TORTUGA

El més popular i conegut del Logo és la seva tortuga. Aquest èxit, malauradament, ha fet oblidar altres aspectes interessants del llenguatge. El Logo va néixer sense la tortuga, però el cert és que ara no el podem imaginar sense ella.

La tortuga es troba inicialment al centre de la finestra de Gràfics, orientada cap amunt. Treballar amb la tortuga consisteix a fer-la moure mitjançant unes ordres de desplaçament i gir. Aquestes ordres són sempre relatives a la posició i l'orientació de la tortuga.

Abans de governar la tortuga directament a través d'ordres del llenguatge Logo, ho fareu amb l'ajuda d'un micromón, especialment pensat pels més petits o per qualsevol novell en Logo, com ara ho sou vosaltres.



Que és un micromón? us estareu preguntant. Un *micromón* és un entorn d'aprenentatge limitat i ben definit en el que succeixen coses interessants. Quan interaccionem amb aquest micromón, aprenem i descobrim idees importants.

Prelogo

La tortuga està tan identificada amb el Logo que és comú denominar "prelogo" al conjunt de micromons que introdueixen el treball amb la tortuga. Aquestes pràctiques no sempre necessiten dels ordinadors, i quan ho fan, poden fer servir per triar les ordres el teclat o altres dispositius com la tauleta sensible.

Vivenciació

Els conceptes geomètrics que la tortuga suggereix poden ser treballats sense l'ordinador, i amb els infants més petits. No penseu que aquests exercicis són tan sols un joc. Simplement mostren com l'anomenada *geometria de la tortuga,* que també té una formalització matemàtica rigorosa, es pot presentar de forma subjectiva.

Són molt interessants els treballs de vivenciació, en els que a traves d'ordres verbals es guia a un suposat robot a través de l'habitació. Les rajoles del terra poden determinar la mida de les passes, i els girs poden ser sempre d'un quart de volta. En principi, les ordres poden ser tan sols "avança" i "gira a la dreta", i després s'hi poden afegir "recula" i "gira a l'esquerra". Finalment es pot afegir un nombre de passes a les ordres de desplaçament, i fer servir girs diversos, com mitja volta, un terç de volta, etc.



Guieu a un dels vostres companys o companyes des de la porta de la classe fins la seva cadira, fent servir tan sols les ordres "avança" i "gira a la dreta". Si s'equivoca o no vol obeir, repetiu el joc tapant-li els ulls amb un mocador. Discutiu diverses ampliacions i "dramatitzacions" d'aquest joc (joc per equips, cercar un tresor, etc).

Tortuga de terra

Encara que amb poca presència al nostre país, la tortuga de terra ha estat el centre de treballs molt interessants realitzats al Regne Unit i a altres països. Aquesta tortuga no és més que una joguina mòbil amb forma de tortuga o de robot, que pot desplaçar-se seguint les ordres que li podem programar mitjançant un petit teclat que porta incorporat, similar al d'una calculadora.

Amb aquesta joguina pren relleu el concepte de "programa", entès com la sèrie d'ordres que guien la tortuga al llarg d'un recorregut complet.

Una alternativa a la tortuga mecànica programable pot trobar-se en la manipulació directa d'una tortuga de joguina. En aquest cas, per fer efectives les ordres, cal agafar la tortuga amb la mà a fi de moure-la o girar-la.

Tortuga gràfica

També disposem de programes interactius, que a través del teclat, permeten treballar amb la tortuga que hi ha a la pantalla de l'ordinador. Aquests programes estan realitzats amb el propi llenguatge Logo, i són un exemple del tipus de programes que qualsevol usuari del Logo amb experiència pot realitzar.

El fitxer PRELOGO.LOG, present al disquet que acompanya aquest manual, conté un d'aquest programes. Per fer-lo servir, heu de saber com recuperar el seu contingut a l'àrea de treball del Logo. Les instruccions següents, que heu de teclejar sense escriure l'interrogant, donen inici al micromón Prelogo.

```
?recupera "a:prelogo
?prelogo
```

Heu d'escriure les instruccions dins la finestra de Treball, a continuació de l'interrogant que fa de símbol d'atenció. Per informar al Logo que heu completat la instrucció, heu de prémer la tecla RETORN. Sempre heu de finalitzar l'escriptura de les instruccions amb aquesta tecla. No ho oblideu, ja que no tornarem a fer-ne esment.

A la primera instrucció es fa servir l'ordre `recupera`, al costat de la qual es troba, precedit de cometes, el nom de fitxer que volem recuperar. Al vostre ordinador aquest nom podria ser una mica diferent al de la instrucció. Per exemple, podeu tenir el disquet amb els exemples introduït a la unitat de disquets B:, i no a la unitat A:. En cas de dubte, demaneu consell. Observeu que no cal teclejar l'extensió del fitxer si aquesta és .LOG.

La instrucció de la segona línia és la que inicia el treball amb el micromón. L'ordre `prelogo` no forma part del conjunt d'ordres disponibles en arrencar el WIN-LOGO. Les instruccions que hi ha al fitxer PRELOGO.LOG són les que defineixen, entre altres, aquesta nova ordre.



Ara per ara, aquestes dues instruccions són per vosaltres com conjurs màgics. Teclegeu-les sense por i, si us equivoqueu, no patiu. Torneu-les a teclejar fins que funcionin, i recordeu que equivocar-se i acceptar els propis errors com etapes del procés d'aprenentatge són parts susbtancials de la filosofia educativa del Logo.

A partir del moment en que teclegeu l'ordre `prelogo`, canvien les regles del joc. Aquestes estableixen el protocol amb el què cal treballar amb el micromón.

Protocol del micromón Prelogo

Podeu demanar a la tortuga que avanci, reculi, giri a la dreta o a l'esquerra un quart de volta, o esborri el dibuix i torni a la posició inicial. Per fer-ho, disposeu de les tecles A, R, D, E, i I, lletres inicials de les accions mencionades. Per donar fi al treball amb el micromón, heu de prémer la tecla F.

Per executar una ordre heu de prémer tan sols una tecla, sense fer servir a continuació la tecla RETORN. Si ho desitgeu, poder fer servir alternativament les tecles de desplaçament, tal com la taula següent mostra.

Taula 1-1. Protocol del micromón Prelogo

Acció	*Inicial*	*Tecla alternativa*
avançar	A	FLETXA AMUNT
recular	R	FLETXA AVALL
girar a la dreta	D	FLETXA DRETA
girar a l'esquerra	E	FLETXA ESQUERRA
iniciar nou dibuix	I	INICI
sortir del micromón	F	FI

Amb aquests senzills elements ja podeu experimentar moltes de les possibilitats d'exploració de l'espai que el Logo ofereix.



Dibuixeu un quadrat; diversos quadrats un dintre de l'altre; sanefes; la façana d'un castell; etc. Podeu descobrir moltes coses, per exemple, que conceptes com “amunt” o “a sota” perden el seu sentit tradicional en treballar amb la tortuga. Que més podeu descobrir?

La tauleta sensible

En lloc del teclat es poden fer servir altres dispositius per trametre les instruccions al Logo. De forma singular, dintre d'aquests medis alternatius destaca la tauleta sensible. Encara que no és present a les escoles de forma generalitzada, tots els centres d'ensenyament primari de Catalunya que han estat dotats d'aules informàtiques pel departament d'Ensenyament de la Generalitat de Catalunya en disposen.

La tauleta sensible pot facilitar l'accés als recursos informàtics als infants no lectors i a les persones amb necessitats educatives especials.

Físicament, la tauleta té una superfície quadrada de 32 centímetres, i és capaç de reconèixer qualsevol contacte a sobre d'ella. Una aplicació informàtica basada en la tauleta consisteix en una làmina dibuixada que es posa a damunt de la tauleta, i els programes d'ordinador que analitzen i processen els contactes que s'hi facin a sobre de la làmina.

El Programa d'Informàtica Educativa ha distribuït diverses aplicacions de la tauleta, i entre elles una relacionada amb el Logo. Aquest material consisteix, apart dels programes, en tres làmines i un llibret informatiu sobre els quatre micromons que s'hi poden treballar. A continuació presentem una breu descripció d'aquests micromons.

Movilogo

Orientat als alumnes d'educació infantil, d'entre 3 i 5 anys, introdueix els recursos més elementals. Treballa bàsicament els moviments absoluts en el pla: avançar, recular, i anar cap a la dreta o l'esquerra. Movilogo

permet introduir el Logo i els processos informàtics bàsics, com ara enregistrar, recuperar, imprimir, etc.

Girlogo

Orientat també als alumnes d'educació infantil, presenta els girs absoluts, fent distinció entre el desplaçament i el gir cap amunt, avall, esquerra i dreta. Encara que la làmina de Girlogo és la mateixa de Movilogo, treu profit d'elements que no s'han treballat abans.

Prelogo

Introdueix la paraula escrita i els procediments, i està destinat a alumnes amb edats compreses entre 4 i 7 anys. També enriqueix l'entorn de treball amb element nous: colors, primitives i girs. Les instruccions, en ser triades damunt la tauleta, s'escriuen automàticament a la pantalla de l'ordinador, i introdueixen així una certa familiaritat amb les ordres escrites.

Minilogo

Destinat a alumnes amb edats compreses entre els 6 i 9 anys, presenta els paràmetres numèrics, la tecla executiva i l'ambient de finestres propi del WIN-LOGO. Obliga a donar valors als desplaçaments i els girs, i a validar les ordres amb la tecla executiva.



Si disposeu d'una tauleta sensible no deixeu de fer-la servir. A quines edats creieu que l'alumnat pot treure profit d'aquesta eina? Quina dinàmica de treball us semblaria vàlida si disposéssiu d'una tauleta a la vostra classe? Quan creieu que el teclat de l'ordinador pot substituir a la tauleta? Com valoreu el document "Aplicacions Logo", que descriu els micromons Logo de la tauleta?

Ordres de la tortuga

Les ordres primitives fonamentals per guiar la tortuga li permeten modificar la seva posició i la seva orientació.



Parlem d'*ordres primitives* perque són ordres que el WIN-LOGO ja porta definides, llestes per ser usades. Vosaltres sabreu aviat com definir ordres noves, a les quals denominarem *procediments* per distingir-les de les primitives. Podeu imaginar les ordres primitives com procediments predefinits.

Desplaçament

Amb poques ordres podeu guiar la tortuga i fer-la dibuixar amb el llapis que porta. Perquè camini, heu de fer servir les primitives `avança` (`av`) i `recula` (`re`). Aquestes dues ordres esperen un paràmetre numèric com a entrada. Això vol dir que, al seu costat i separat per un espai, heu d'escriure el nombre de passes que la tortuga ha de caminar. No està definida la correspondència entre la mida d'una passa de tortuga i la mida física de la pantalla de l'ordinador.

Sol resultar pràctic acabar els dibuixos deixant a la tortuga al mateix lloc on es trobava abans de començar a dibuixar. Proveu de teclejar les instruccions següents:

```
?avança 50
?recula 100
?avança 50
```

S'ha dibuixat una recta i la tortuga ha tornat a la posició inicial. Traçar una nova recta, perpendicular a la que ja tenim, donaria lloc a una creu. Per dibuixar-la, però, cal que feu girar la tortuga.

Gir

Per canviar la orientació de la tortuga heu de recórrer a les primitives `gira.dreta` (gd) i `gira.esquerra` (ge). Aquestes dues ordres tenen com a entrada un paràmetre numèric: les unitats a girar. La magnitud del gir s'ha d'expressar en graus. La tortuga sap girar sobre si mateixa, i divideix el gir en petites unitats. Quan la tortuga gira un total de 360 d'aquestes unitats —els graus—, torna a l'orientació inicial. Si 360° són un gir complet, 90° han de ser un quart de volta, el gir necessari per posar a la tortuga en posició horitzontal. Per dibuixar la recta perpendicular que ens feia falta i completar la creu, teclegeu les instruccions següents:

```
?gira.dreta 90
?recula 50
?avança 100
?recula 50
?gira.esquerra 90
```

Ara que la creu està dibuixada podeu amagar la tortuga. Amb l'ordre `desapareix` (dap) podeu fer que la tortuga deixi de ser visible. Si la voleu veure de nou, executeu la primitiva `apareix` (ap).

El llapis

Si la tortuga deixa rastre en desplaçar-se és per que porta amb ella un llapis. La tortuga pot deixar estar el llapis i moure's sense dibuixar, o canviar-lo per un altre llapis amb mida o color diferent.

Per dibuixar una altra creu igual, a 150 passes a la dreta de la que ja està dibuixada, i separada d'aquesta, heu de demanar a la tortuga que deixi estar el llapis un moment, amb la primitiva `no.llapis`. Per tornar a dibuixar, heu

de fer servir la primitiva `llapis`. Les instruccions següents faran que la tortuga es desplaci fins al punt desitjat sense deixar rastre. Teclegeu-les dins la finestra de Treball.

```
?no.llapis
?gira.dreta 90
?avança 150
?gira.esquerra 90
?llapis
```

Ara podeu repetir les instruccions que dibuixen la creu. No cal que les torneu a teclejar, ja que la finestra de Treball conserva totes les instruccions que heu teclejat amb anterioritat.



Podeu recuperar les instruccions velles servint-vos del ratolí o de les tecles de desplaçament. Exploreu sistemàticament les possibilitats d'edició que dins la finestra de Treball us ofereixen les tecles FLETXA AMUNT, FLETXA AVALL, PÀGINA AMUNT, INICI, etc.

La finestra de Gràfics



Si voleu netejar la finestra de Gràfics i retornar la tortuga a la seva posició i orientació originals, heu d'activar la finestra i triar la comanda **Neteja** al menú **Finestra**. Podeu obtenir el mateix resultat si teclegeu l'ordre `inicia.dibuix` (id).

Per desar els vostres dibuixos en un fitxer o per recuperar-los i tornar a veure'ls, el menú **Finestra** us ofereix les comandes **Desa** i **Recupera**. Per executar-les activeu la finestra de Gràfics i trieu-les del menú. Amb aquesta tècnica podeu desar al disc dibuixos d'elaboració molt lenta, per recuperar-los després en un instant.

Sumari

A cada capítol d'aquest manual s'introduiran noves primitives. Les aparegudes en aquest primer capítol, i recollides a la taula següent, afecten totes a la tortuga, fora de la primitiva `recupera`. També s'inclouen a la taula les primitives gràfiques que seran mencionades més endavant dins l'apartat "Material complementari". No oblideu consultar la definició completa de totes aquestes primitives amb el sistema d'ajuda del WIN-LOGO.

Taula 1-2. Primitives tractades

Nom	*Entrades*	*Efecte*
`apareix`	0	La tortuga apareix
`avança`	1	La tortuga avança
`desapareix`	0	La tortuga s'amaga
`fes.color`	1	Selecciona el color del llapis
`fes.gruix`	1	Selecciona el gruix del llapis
`gira.dreta`	1	La tortuga gira cap a la dreta
`gira.esquerra`	1	La tortuga gira cap a l'esquerra
`goma`	0	La tortuga agafa la goma d'esborrar
`inicia.dibuix`	0	La tortuga torna a l'estat inicial
`llapis`	0	La tortuga agafa el llapis
`no.llapis`	0	La tortuga deixa el llapis
`omple`	0	Omple de color un dibuix tancat
`recula`	1	La tortuga retrocedeix
`recupera`	1	Processa un fitxer amb instruccions

La tortuga és l'aspecte més popular i conegut del Logo, però no és l'únic.

Podem governar la tortuga amb gran facilitat, gràcies a micromons especials.

Anomenem ordres de la tortuga a les primitives que fan moure o girar la tortuga, o que modifiquen el seu estat.

Exercicis



Ja disposeu d'elements suficients per explorar el món de la tortuga. Investigueu aquest món fent els exercicis següents, i no menystingueu la importància de l'últim exercici.

Exercici 1-1. Mesureu l'amplada i l'alçada de l'àrea visible a la finestra de Gràfics.

Exercici 1-2. Dibuixeu un quadrat que tingui per centre la posició inicial de la tortuga.

Exercici 1-3. Feu un gran quadrat de 10 passes per dintre de les voreres de la finestra de Gràfics.

Exercici 1-4. Dibuixeu uns eixos de coordenades, en tota la longitud i amplada de la finestra de Gràfics.

Exercici 1-5. Dibuixeu un triangle qualsevol.

Exercici 1-6. Plantegeu un exercici, similar als anteriors, als vostres companys i companyes. Qui plantegi un exercici que ningú pugui resoldre, paga penyora!

Material complementari



Aquest apartat sobre material complementari, present a tots els capítols, us el podeu saltar lliurement si aquest curs és el primer que feu sobre el Logo. Si ja teniu experiència o apreneu depressa, trobareu entreteniment per tota la setmana.

Gruix i color del llapis

El llapis que porta la tortuga pot canviar de mida. La primitiva `fes.gruix` necessita un paràmetre numèric amb un valor entre 1 i 10. `fes.color` espera també un paràmetre numèric, un codi de color amb valor entre 1 i el nombre màxim de colors que accepti la pantalla de l'ordinador amb que treballeu. Experimenteu amb aquestes primitives, i afegiu variació als exercicis del capítol.

En relació al llapis, la tortuga té una altra propietat interessant: pot deixar el llapis i canviar-lo per una goma d'esborrar. Per demanar-li a la tortuga que canviï el llapis per la goma heu de cridar a la primitiva `goma`. Alerta! Després de cridar a `goma` la tortuga esborra per allà on passa. Les primitives `llapis` o `no.llapis` anul·len aquest comportament de la tortuga.

Si feu entrar la tortuga dins d'una figura tancada amb el llapis aixecat, i aleshores el baixeu i crideu a la primitiva `omple`, la figura s'omplirà amb el color del llapis. Molt important: el color del llapis ha de ser el mateix que el de les voreres de la figura que voleu omplir.

Exercici 1-7. Amb quants colors podeu treballar en el vostre ordinador? Que passa si crideu a `fes.color` amb un codi de color incorrecte?

Exercici 1-8. Dibuixeu les quatre barres de la bandera.

Exercici 1-9. Dibuixeu la façana d'una casa, amb porta i finestra de colors diferents.

Adaptació del micromón Prelogo

Podeu adaptar el micromón, modificant la mida del desplaçament i l'angle de gir. Per fer-ho, heu de canviar el contingut de dues paraules que el micromón fa servir: `passa` i `gir`. La primitiva `posa.a`, que serà tractada al proper capítol amb tot detall, serveix per canviar el contingut d'una paraula. Així, per establir el gir a 45°, heu de teclejar l'ordre següent.

```
?posa.a "gir 45
```

Per modificar la mida dels desplaçaments heu de posar un nou valor a la paraula `passa`, que inicialment és 20. Per obtenir una precisió més gran, executeu l'ordre següent.

```
?posa.a "passa 5
```

Si ara torneu al micromón Prelogo, teclejant l'ordre `prelogo`, podreu dibuixar més coses que quadrats i angles rectes.

Exercici 1-10. Quines figures geomètriques es poden dibuixar amb girs de 45°? I amb girs de 60°?

Exercici 1-11. Quins valors de gir i desplaçament us permetrien dibuixar una circumferència?

Exercici 1-12. Dibuixeu un circuit de "carreres" per després obligar a la tortuga a seguir-lo. Deseu la imatge del circuit en un fitxer de nom CIRCUIT.IM, i recupereu la imatge cada vegada que hagueu d'iniciar el recorregut.

2

INSTRUCCIONS

L'objectiu d'aquest capítol és explicar la noció d'instrucció que fa servir el Logo, encara que sense l'afany d'ésser estrictament rigorosos en el tractament del tema. De totes formes, es descriuran exhaustivament tots els elements que poden aparèixer a les instruccions, fet que us pot produir un cert empatx inicial.



No us preocupeu per assimilar tot el contingut del capítol d'una vegada. Us el podeu administrar en petites dosis, tornant més endavant als apartats que ara no entengueu del tot.

Reflexioneu sobre les dificultats d'aprenentatge que se presentin, i de quina forma les supereu. Les tècniques que ara apliqueu us hauran de servir quan feu servir el Logo a la vostra classe. Penseu en els vostres problemes, ara que apreneu Logo, i en els que penseu que podeu tenir quan l'ensenyeu i el feu servir a classe. És molt probable que la temporització d'aquest capítol no us sembli la millor per aplicar en el nivell educatiu en què treballeu. Com organitzaríeu el material? Que deixaríeu per més endavant?

L'intèrpret



Una *instrucció* dins del llenguatge Logo seria equivalent a una oració dins de la llengua catalana: una unitat mínima de significació completa, autònoma sintàcticament. Una instrucció no pot ocupar més d'una línia, encara que dues o més instruccions poden seguir-se dins una mateixa línia.

Quan premeu la tecla RETORN, després d'haver teclejat una instrucció completa, comença el treball de l'intèrpret del llenguatge Logo. És responsabilitat seva analitzar el que heu escrit, executant les ordres que heu teclejat. Si heu comès alguna falta d'ortografia o de sintaxi, la instrucció no podrà executar-se.

A mida que treballeu amb instruccions més complexes, augmenta la probabilitat que cometeu errors d'escriptura. Quan això succeeix, el WIN-LOGO intenta situar l'origen de l'error —no sempre amb massa èxit— i escriu un missatge descriptiu.



Aquest missatge descrivint l'error pot aparèixer a la finestra de text o dins d'un quadre de diàleg, segons si està activada o no l'opció **Desactiva finestra d'error** del menú **Sistema**. El comportament per defecte del WIN-LOGO consisteix a escriure dins la finestra de Text, però ara que comenceu, i segurament cometreu errors amb freqüència —cosa completament natural—, us convindria triar l'altre sistema d'informació.

Ordres

Si podem dir que a cada línia de Logo hi ha d'haver com a mínim una instrucció, també podem afirmar que a cada instrucció ha d'haver una ordre —només una— i que

aquesta ordre sempre encapçala la instrucció. Agafant com exemple l'ordre avança, que ja coneixeu, la figura següent disecciona les diverses parts que pot tenir una instrucció.

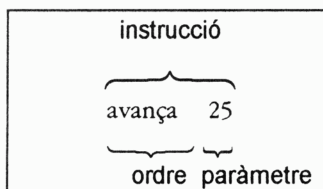


Figura 2-1. Forma general d'una instrucció.

Una ordre es caracteritza fonamentalment per produir un efecte —fer alguna cosa— i tenir un nombre determinat d'entrades. Quan s'executa una ordre ha de rebre un valor per cada una de les seves entrades, si és que en té. Aquests valors són el que denominem paràmetres. Així, seguint la figura anterior, direm que avança té una entrada, i que a l'exemple el seu paràmetre és 25.

Al capítol anterior, i per governar a la tortuga, heu escrit ja unes quantes instruccions amb el llenguatge Logo. Heu vist instruccions formades per ordres que no requereixen cap entrada, com ara apareix, i compostes per tant, d'una única paraula. Altres ordres sí que necessiten entrades, com per exemple avança, i s'han d'escriure amb els seus paràmetres al costat. Les instruccions resultants es componen del nom de l'ordre —una paraula— i dels paràmetres que aquesta ordre pren com a entrada.

Totes les ordres que heu après fins ara, o bé no tenen cap entrada, o només en tenen una, sempre de caire numèric.

Més endavant coneixereu ordres que tenen diverses entrades, i també altres valors, diferents dels numèrics, que el Logo pot tractar.

Expressions

Les següents instruccions són completament equivalents, i si les teclegeu produiran un desplaçament idèntic de la tortuga. Comproveu-ho.

```
?avança 50
?avança 25 + 25
?avança 25 * 2
```

La diferència entre les tres instruccions es troba en la forma en que està expressat el valor que avança rep com a entrada. En el primer cas, el que hi ha escrit al costat d'avança és una constant, i en els altres és una expressió que combina dues constants i un operador. El valor resultant és sempre el mateix, pel que des del punt de vista d'avança no hi ha cap diferència. En el segon i tercer casos, però, per obtenir el valor que ha de rebre avança s'ha d'avaluar una expressió, realitzant els càlculs adequats. De forma general, podem dir que en qualsevol lloc on hi a d'haver un valor, s'hi pot trobar una expressió.



Entenem per *constant* un valor que està escrit de forma explícita, completa, disponible sense necessitat de càlculs complementaris. Una *expressió* és una seqüència formada per constants, variables, funcions, operadors i parelles de parèntesis, de manera que en ser avaluada proporciona un valor.

Aviat explicarem la definició anterior amb detall, però abans d'entrar en matèria, i per facilitar la comprensió del

que serà exposat, presentarem una nova primitiva. Es tracta de `mostra`, una ordre amb una entrada, i que té per efecte mostrar el seu paràmetre d'entrada dins la finestra de text. D'aquesta manera, podreu veure el resultat dels càlculs aritmètics, i comprovar visualment el seu funcionament. Si repetiu les instruccions anteriors, però amb l'ordre `avança` canviada per `mostra`, comprovareu que el valor del paràmetre d'entrada de l'ordre és sempre 50. Teclegeu les instruccions següents. Dins de quina finestra escriu l'ordre `mostra`?

```
?mostra 50
 50
?mostra 25 + 25
 50
?mostra 25 * 2
 50
```

Nombres

En el llenguatge Logo, les constants numèriques, o simplement nombres, estan constituïts per una sèrie de dígits, precedits del signe menys (-) en el cas dels nombres negatius. Si el nombre té part decimal, aquesta està separada de la part entera per un punt, com també es fa a les calculadores. Teclegeu els següents exemples:

```
?mostra 1992
 1992
?mostra -1
 -1
?mostra 3.141592
 3.141592
```

El Logo també coneix l'anomenada notació exponencial o científica. Si necessiteu fer servir aquesta notació ja sabreu com funciona. Si no es així, és que no us cal conèixer-la.

De totes formes, i per si teniu curiositat, els exemples següents parlen per si mateixos (?).

```
?mostra 1e2
 100
?mostra 1e-2
 0.01
```

Operadors

Els operadors aritmètics, que són els que presentarem en aquest capítol, permeten combinar dos nombres per obtenir-ne un de nou. Tots els operadors són binaris, el que vol dir que necessiten dos operands per fer la seva feina, i accepten notació prefixa i infixa. En la pràctica fareu servir sempre la notació infixa, és a dir, amb l'operador situat enmig dels operands. El caràcter menys (–), amb notació prefixa, també pot actuar com l'operador de negació.

En el llenguatge Logo els diferents operadors es representen a traves d'un únic caràcter, d'una forma similar a molts altres llenguatges de programació. Pel que fa a la suma (+) i a la resta (–), els caràcters són els que us podíeu imaginar. La multiplicació (*) i la divisió (/) ja no segueixen la simbologia tradicional, encara que en el món informàtic està tan estès l'ús de l'asterisc i la barra inclinada, que resulten familiars a qualsevol que hagi treballat amb ordinadors. Teclegeu els exemples següents:

```
?mostra 2 + 2
 4
?mostra 2 - 2
 0
?mostra 2 * 2
 4
?mostra 2 / 2
 1
```

El Logo, de forma excepcional, permet que els operadors "toquin" els valors sobre els que han d'operar. Per fer això possible, el llenguatge ha de considerar els caràcters que representen les operacions de forma especial. Els exemples anteriors es poden escriure, doncs, de la forma següent.

```
?mostra 2+2
 4
?mostra 2-2
 0
?mostra 2*2
 4
?mostra 2/2
 1
```

El resultat és el mateix, i tan sols canvia la presentació. Dependrà del context o dels gustos personals, que feu servir espais o no per separar els operadors, però us heu d'acostumar a acceptar i entendre les dues notacions. Alerta, però, amb el caràcter menys (-). A l'exemple següent, el Logo el considera com l'operador de negació, pel que la instrucció no s'executa potser com es voldria. Podeu explicar el missatge d'error que el WIN-LOGO produeix?

```
?mostra 2 -2
 2
 No sé què s'ha de fer amb -2
```

Ordre d'avaluació

Quan en una expressió intervenen diversos operadors, aquests s'avaluen en un ordre particular. Primer la multiplicació i la divisió, i en segon lloc la suma i la resta. Si voleu modificar aquest ordre, podeu fer servir lliurement parelles de parèntesis, per agrupar les

operacions que s'han de realitzar primer. Comproveu la importància dels parèntesis teclejant els exemples següents.

```
?mostra 2 * 2 + 3
 7
?mostra 2 * (2 + 3)
 10
```

Quan a l'hora d'escriure una expressió complexa dubteu sobre l'ordre en què s'avaluaran els seus elements, no us ho penseu: poseu parèntesis, i endavant!

Representació interna dels nombres



Ara ja disposeu dels elements fonamentals per fer servir el Logo com una calculadora elemental. Però el Logo, com qualsevol calculadora, té els seus límits. Trobeu-los! Quin és el nombre més gran, o més petit, que el Logo accepta? Què passa si demaneu al Logo que mostri un nombre format per setze nous (9999999999999999), o per un nou amb nou decimals (9.999999999)? Algú de vosaltres pot donar explicació a aquesta pèrdua de precisió?

Funcions

Si bé els operadors aritmètics del Logo resolen les quatre operacions més habituals amb nombres, hi ha altres operacions d'ús comú. Per exemple, la potenciació o l'arrel quadrada. El Logo ens ofereix aquestes operacions a través de les funcions.

Les funcions tenen una sintaxi força similar a les ordres. Tenen un nom —una paraula—, poden necessitar paràmetres d'entrada, però a diferència de les ordres, retornen un valor, o dit d'una altra manera, tenen una

sortida. D'aquesta forma, en qualsevol lloc on hi hagi d'haver un valor, pot posar-se una funció amb els seus corresponents paràmetres. Com sempre, teclejar un parell d'instruccions aclarirà aquests conceptes. Comencem amb la funció `arrel`. Com és previsible, espera un nombre com entrada, i ens retorna l'arrel quadrada d'aquest nombre.

```
?mostra arrel 25
 5
?mostra arrel 2
 1.41421356
```

Calcular la potència d'un nombre és també una operació habitual. La funció `potència` té dues entrades. En primer lloc, la base, i en segon l'exponent. És a dir, primer, el nombre del qual volem calcular una potència i després la potència en qüestió. Elevar un nombre al quadrat pot no necessitar de la funció `potència`, però està clar que tots els altres casos si que ho necessiten.

```
?mostra 5 * 5
 25
?mostra potència 5 2
 25
?mostra potència 5 3
 125
```

De les funcions diem que tenen notació prefixa. Això vol dir que primer s'escriu el nom de la funció, i després els seus paràmetres. És una situació diferent a la dels operadors, els quals sempre fem servir amb notació infixa.

El teorema de Pitàgores

Ara, aprofitant que podem calcular l'arrel quadrada d'un nombre, anem a aplicar el popular teorema de Pitàgores

per dibuixar un quadrat i les seves diagonals. Recordeu: el quadrat de la longitud de la hipotenusa d'un triangle rectangle és igual a la suma dels quadrats dels altres costats.

En primer lloc dibuixeu el quadrat. Feu executar quatre vegades la línia següent (escrivint-la només una vegada).

```
?avança 50 gira.dreta 90
```

Per fer la diagonal, primer heu d'orientar la tortuga, i després fer-la avançar recorrent la longitud de la hipotenusa.

```
?gira.dreta 45
?avança arrel 50*50 + 50*50
```

Ara col·loqueu la tortuga en el vèrtex inferior dret.

```
?gira.dreta 45+90
?avança 50
```

Orienteu la tortuga adequadament i feu-la avançar a través de la hipotenusa.

```
?gira.dreta 45+90
?avança arrel 50*50 + 50*50
```

I finalment, és de bona educació deixar la tortuga on l'heu trobat.

```
?gira.dreta 45
?recula 50
```

Ja està! Si trobeu punts foscos en les operacions anteriors, repetiu-les fins que entengueu tant la sintaxi del llenguatge com els procediments geomètrics emprats.

Nombres enters i reals

El concepte de nombre del Logo es correspon alhora amb dues abstraccions numèriques matemàtiques: els nombres enters i reals. Generalment, no us heu de preocupar d'aquest tema, però hi ha ocasions en què us caldrà fer-ho. Per exemple, a l'hora de fer divisions es produeix un fet que no passa amb cap altre operador aritmètic. Si dividim dos nombres enters —sense decimals—, el resultat por ser un altre nombre enter o real, segons si els nombres són o no divisibles entre si. Les altres operacions —suma, resta i multiplicació— només donen lloc a nombres reals si algun dels seus operands ja ho és.

```
?mostra 4 / 2
 2
?mostra 4 / 3
 1.33333333
```

Hi ha ocasions en què podeu necessitar saber el quocient sencer entre dos nombres, deixant de banda els decimals, i també us pot fer falta conèixer el residu d'una divisió entera. Per calcular aquests valors, el Logo proporciona les funcions `quocient` i `residu`. Totes dues esperen dos paràmetres, el dividend i el divisor amb què operar. Per altra banda, si necessiteu convertir un nombre real en enter, és a dir, si voleu eliminar els decimals, podeu fer servir la funció `part.entera`.

Teclegeu les instruccions següents, i altres que vosaltres decidiu, fins que veieu clar el funcionament d'aquestes funcions.

```
?mostra quocient 13 4
 3
?mostra residu 13 4
 1
?mostra part.entera 3.1416
 3
?mostra part.entera 4/3
 1
```



Avui en dia, en un món ple de calculadores que només coneixen els nombres reals, creieu convenient continuar parlant de nombres enters o racionals? Creieu que cal saber operar amb fraccions? Encara més, penseu que cal saber resoldre manualment operacions com ara divisions amb decimals o arrels quadrades?

Variables

Si fins ara necessitàveu calcular un valor intermedi, per fer servir en un càlcul posterior, podíeu fer escriure el resultat del primer càlcul a la finestra de Text, per tal de llegir-lo més endavant i copiar-lo manualment en la nova expressió. Si bè aquesta tècnica pot ser vàlida en algunes ocasions, el Logo ofereix un mètode més poderós i general per realitzar la mateixa tasca —emmagatzemar temporalment un valor—

Una paraula pot tenir associada una ordre o una funció, ja sigui primitiva o no, i quan el Logo troba una paraula dins d'una instrucció així ho considera. Paral·lelament, una paraula pot tenir associat un contingut, per exemple, un valor numèric. Per diferenciar entre els dos tipus de valors que poden estar associats a una paraula, el Logo ha de recórrer a les cometes ("). Si una paraula no té al davant unes cometes, el Logo considera que és una ordre o funció, i intenta executar-la. Per evitar aquest

comportament, cal precedir la paraula per unes cometes. Uns exemples i una nova primitiva us aclarirà aquest embolic.

```
?mostra color
 1
?mostra "color
 color
```

La funció `color` retorna el color del llapis de la tortuga, que en el vostre ordinador pot ser diferent al del mostrat en l'exemple. La primera instrucció de l'exemple dóna lloc a l'escriptura del valor retornat per la funció `color`, però la segona mostra la pròpia paraula `color`, cosa molt diferent.

Per assignar un valor al contingut d'una paraula, el Logo proporciona l'ordre `posa.a`, i per accedir al contingut la funció `contingut`. Comproveu el funcionament d'aquestes primitives teclejant les instruccions següents:

```
?posa.a "color "vermell
?mostra contingut "color
 vermell
```

Un recurs habitual per fer dibuixos en escales diferents consisteix a basar els desplaçaments en el contingut d'una variable. Teclegeu les instruccions següents:

```
?posa.a "passa 25
?avança contingut "passa
```

L'expressió `contingut "`*`variable`* és tan habitual que el Logo ofereix una alternativa abreujada. El caràcter dos punts (`:`) és equivalent al conjunt format per la paraula `contingut` i les cometes. Així, les dues instruccions següents són equivalents. Comproveu-ho teclejant-les dins la finestra de Treball.

```
?mostra contingut "color
 vermell
?mostra :color
 vermell
```



Els conceptes anteriors són la medul·la del llenguatge Logo, i tornarem a ells novament. Si els hem introduït en aquest capítol és per completar la descripció dels elements que poden intervenir en les expressions. No us preocupeu si ara us queden punts foscos. És normal, i us caldrà tornar repetides vegades sobres aquestes idees fins que les assimileu per complet.

Sumari

El conjunt d'ordres, funcions i operadors predefinits, és el que coneixem com a *primitives* del llenguatge Logo.

En una expressió hi poden intervenir constants, operadors, funcions i variables. L'ordre d'avaluació pot modificar-se amb parelles de parèntesis.

Els operadors aritmètics implementen les quatre operacions artimètiques fonamentals. Les altres operacions es realitzen a través de funcions.

Taula 2-1. Operadors aritmètics

Caràcter	*Operació*
+	suma
–	resta
*	multiplicació
/	divisió

Les paraules tenen un contingut al qual es pot accedir. Aquesta és la forma en que el Logo permet implementar el concepte de variable, comú a molts llenguatges de programació.

Taula 2-2. Primitives tractades

Nom	*Entrades*	*Efecte*
`arrel`	1	Retorna l'arrel quadrada
`color`	0	Retorna el color del llapis
`contingut`	1	Retorna el contingut d'una paraula
`mostra`	1	Mostra la seva entrada com text
`part.entera`	1	Trunca un nombre real
`posa.a`	2	Assigna un valor a una paraula
`potència`	2	Retorna la potència d'un nombre
`quocient`	2	Retorna el quocient d'una divisió
`residu`	2	Retorna el residu d'una divisió

Exercicis



L'èmfasi que en aquest capítol hem posat en els nombres no us ha de fer pensar que el Logo està especialment adaptat pel seu tractament. Si ho hem fet, és perquè els nombres ens han permès descriure, a través d'un material familiar, tots els elements que poden intervenir en les expressions.

Si no us interessen massa les matemàtiques no us amoïneu. Contràriament a la que és una opinió generalitzada, la programació d'ordinadors no té massa a veure amb les matemàtiques. De fet, el que manca al món

de la informàtica és gent de lletres. Discutiu aquesta opinió!

Exercici 2-1. Feu una predicció sobre el resultat d'avaluar les expressions aritmètiques incloses en les instruccions següents. Comproveu la vostra hipòtesi teclejant les instruccions.

a. `mostra 2 * 3 + 2`

b. `mostra 2 * (3 + 2)`

c. `mostra 2 * 3 / 2`

d. `mostra 2 * (3 / 2)`

Exercici 2-2. Traduïu les expressions aritmètiques següents al llenguatge Logo. Escriviu el seu resultat amb l'ajuda de la primitiva `mostra`.

a. $-5 + -4 + 7 + 8$

b. $5 + 3 \times 2$

c. $\frac{2}{3} \times \frac{1}{5} + 2$

Exercici 2-3. Apliqueu novament el teorema de Pitàgores per dibuixar un quadrat i les seves diagonals. Ara, però, feu servir la funció `potència` per calcular els quadrats dels costats.

Exercici 2-4. Coneixent la igualtat $\sqrt[n]{m} = m^{1/n}$, calculeu el valor de les expressions següents. Com que el Logo no disposa d'una funció específica per calcular l'arrel enèsima d'un nombre, haureu de fer servir la primitiva `potència` per calcular el resultat. Fixeu-vos que, per exemple, $\sqrt[2]{4}$ és igual a $4^{1/2}$.

a. $\sqrt[6]{81}$

b. $\sqrt[3]{3^2}$

c. $\sqrt[12]{2}$

Exercici 2-5. Tornem al teorema de Pitàgores. Dibuixeu un rectangle de base 100 i alçada 50. Ara, afegiu al dibuix un triangle isòsceles inscrit dins del rectangle. Penseu que aquest triangle haurà de tenir dos angles de 45° i un de 90°.

Exercici 2-6. Coneixent la igualtat dividend = divisor x quocient + residu, que recull tots els elements que intervenen en una divisió entre nombres enters, calculeu els valors que falten en les expressions següents. Demaneu al Logo que calculi els valors exactes.

a. quocient x 2 + residu = 15

b. 8 x 2 + residu = 17

c. quocient x 7 + 3 = 17

d. 2 x divisor + 1 = 13

Material complementari



Recordeu que en aquests exercicis s'introdueixen a vegades nous materials, que seran tractats en capítols posteriors. Feu aquests exercicis només si teniu ganes d'avançar matèria, i si no els podeu resoldre ara, penseu a tornar més endavant, quan els conceptes involucrats hagin estat tractats.

Exercici 2-7. Quina diferència trobeu entre les ordres i les funcions? En què són similars?

Exercici 2-8. Repetiu el dibuix proposat a l'exercici 2-5, però de la forma següent. Dibuixeu un quadrat i una de les seves diagonals, i al seu costat i enganxat, un nou quadrat dibuixant també la diagonal simètrica a la de l'altre quadrat. Una vegada fet això, al dibuix li "sobra" la línia que segueix l'alçada del triangle. Esborreu-la amb l'ajuda de la primitiva goma.

Exercici 2-9. Poseu els parèntesis necessaris a les expressions següents perquè la igualtat sigui correcta. Comproveu la vostra proposta demanant al Logo que avaluï les expressions. Si escriviu les expressions tal com apareixen aquí, incloent-hi el signe d'igualtat, què mostra el WIN-LOGO a la finestra de Text? Quina explicació podeu donar d'aquest fet?

a. `5 * 3 + 2 = 25`

b. `5 * 3 + 2 - 14 / 2 = 9`

c. `5 * 3 + 2 - 14 / 2 = -10`

Exercici 2-10. Repetiu el dibuix d'un quadrat i les seves diagonals, però sense fer servir cap nombre, tan sols ordres i variables. Executeu abans les instruccions següents.

```
?posa.a "costat 50
?posa.a "gir 90
?posa.a "mig.gir 45
?posa.a "diagonal arrel 50*50*2
```

Exercici 2-11. Enumereu i definiu amb precisió els diferents elements que poden intervenir en una instrucció.

Exercici 2-12. Sabent que el residu resultant de dividir un nombre parell entre dos és sempre zero, demaneu al

Logo si els nombres següents són parells o senars. Sabríeu fer que el Logo contesti, no escrivint el residu, sinó les paraules `ver` i `fals`?

a. 2

b. 123

c. 1992

d. 11219

3

PROCEDIMENTS

Segur que a aquestes alçades ja us heu cansat d'escriure instruccions del Logo, una darrera l'altre. Potser penseu que amb aquesta tècnica no es pot arribar gaire lluny, i teniu raó. Per sort, el que ara trobeu a faltar —i encara més— ho trobareu al Logo. És possible fer un grup amb diverses instruccions, donar-li un nom, i fer servir el nom per executar totes les instruccions d'una vegada.



Aquests grups d'instruccions reben el nom de *procediments*. Un cop definits, els podeu fer servir de la mateixa forma que les ordres o funcions primitives, i els podeu incloure a les instruccions amb què realitzeu altres procediments. Encara que és possible definir ordres i funcions noves, el conjunt d'operadors predefinits no pot ampliar-se. De fer servir el nom d'un procediment dins d'una instrucció en diem *cridar* el procediment.

Definició de procediments

Les primitives que permeten definir procediments són la parella `procediment` i `fi`, que tenen una sintaxi especial, diferent a tot el que hem explicat al segon capítol. Podeu considerar la combinació entre aquestes dues primitives, a les quals no podríem classificar ni com

a ordre ni com a funció, com a una instrucció especial. La definició d'un procediment sempre comença amb la primitiva `procediment` i acaba amb la primitiva `fi`.



Podeu definir nous procediments dins la finestra de Text. Per començar la definició, només heu d'escriure la paraula `procediment` seguida del nom que voleu posar al procediment. Observareu que el símbol d'atenció del Logo, l'interrogant (?), canvia pel caràcter 'major que' (>) quan premeu la tecla RETORN. A partir d'aquest moment s'han d'escriure, ocupant tantes línies com calgui, les instruccions que s'executaran quan es cridi el procediment, i que ara no seran avaluades. Per donar per acabada la definició, heu de teclejar, a la darrera línia del procediment, la primitiva `fi`. Arribats a aquest, punt el WIN-LOGO informa que s'ha definit un nou procediment, escrivint un missatge a la finestra de Text.

Teclegeu les instruccions següents, dins la finestra de Text, per definir i executar el vostre primer procediment. Trobareu una nova primitiva, la funció `atzar`, que retorna un nombre aleatori entre zero i el seu paràmetre menys un. Així, l'expressió `atzar 10` produirà un nombre entre 1 i 9, ambdós inclosos. Aquesta funció té un paper molt important a l'hora de realitzar simulacions amb l'ordinador, i la tornarem a fer servir més endavant.

```
>procediment passeig
>   avança atzar 10
>   gira.dreta atzar 20
>   passeig
>fi
```

Si no heu comès cap error d'escriptura el WIN-LOGO deu haver escrit el missatge `Acabat de definir passeig`

a la finestra de Text. Ara ja podeu cridar al procediment, però abans feu neta la finestra de Gràfics.

```
?inicia.dibuix
?passeig
```

Si les instruccions que conté el procediment no tenen cap error, la tortuga deu estar donant voltes com una boja. Per aturar aquest passeig a l'atzar de la tortuga, heu de prémer la tecla ESC, a la qual sempre podeu recórrer per aturar l'execució d'un procediment.

Recursivitat

Segurament us deu estar preocupant la penúltima línia del procediment. Que fa, deveu pensar, el nom del procediment dins la seva definició? Doncs be, això es pot fer, i rep el nom de recursivitat o recurrència. La possibilitat de definir procediments recursius és una de les característiques més interessants i poderoses del Logo, i també una de les més confuses pels novells en l'ús del llenguatge.

Aquesta dificultat fa que s'amagui, a vegades, aquesta característica del llenguatge als que comencen a estudiar-lo. Com veieu, aquí hem optat per la solució contrària, que és triar com a primer exemple de procediment un de recursiu. Encara esperarem alguns capítols per treballar la recursivitat de forma habitual, però ara ja sabeu que existeix, i us podeu començar a mentalitzar per acceptar-la més endavant amb naturalitat. L'efecte que produeix cridar a `passeig` dins de `passeig` és el mateix que executar repetidament fins l'infinit les instruccions de les dues primeres línies del procediment.

```
?avança atzar 10
?gira.dreta atzar 20
?avança atzar 10
?gira.dreta atzar 20
 ...
 ...
```

Veureu molts procediments recursius en propers capítols, i en cada situació intentarem aclarir les dificultats que es presentin. La principal dificultat, però, l'heu de resoldre vosaltres, i no és cap altra que pensar equivocadament que la recursivitat és difícil d'entendre. La recursivitat és agradable, bonica, excitant, expressiva, poderosa..., i fàcil.

Exemples de recursivitat



La recursivitat és, de fet, un concepte "natural". Podeu trobar exemples de recursivitat a la literatura —destacaria el Quixot—, al cinema —algunes pel·lícules de Fellini—, a la pintura —Velazquez dins el quadre de las Meninas—, la música —Bach, entre d'altres—, etc.

Discutiu els següents exemples de recursivitat extrets de la vida quotidiana, o de situacions o fets que segur que heu observat. Podeu afegir altres exemples? On creieu que es troba l'atractiu que exerceix la recursivitat sobre el nostre pensament?

- El reflex de dos miralls, posats un davant de l'altre.
- Un somni en el que despertem d'un somni en què somniàvem que despertàvem d'un somni en què...
- Un televisor que mostra la imatge d'un televisor que mostra la imatge d'un televisor...

- Un conte en el que un personatge interromp la història per narrar un conte, en què també un personatge narra un conte...

L'editor



Treballar amb procediments porta a introduir el concepte de *programa*. Un programa seria un conjunt de procediments, reunits en un o més fitxers, que col·laboren en una tasca global.

Encara que, com heu vist, els procediments es poden definir dins la finestra de Treball, és molt més convenient treballar els programes dins de la finestra d'Edició. Per obrir-la, heu d'executar la comanda **Edició** del menú **Entorns.** Si ho desitgeu, podeu fer que la finestra ocupi tota la pantalla executant la comanda **Maximitza** del menú **Finestra**.

Dins la finestra d'Edició podeu escriure qualsevol text, no tan sols programes, de forma similar a com ho faríeu amb un editor o processador de textos convencional. El menú **Edició** disposa de les comandes habituals per manipular text, i el menú **Recerca** ofereix, entre altres, comandes per buscar o substituir fragments de text. Estudieu sistemàticament les possibilitats que us ofereixen aquests dos menús. Al menú **Finestra** també trobareu comandes útils, com ara **Neteja,** amb què podeu eliminar el contingut de qualsevol finestra.

Quan escriviu procediments o instruccions nous dins la finestra d'Edició, les instruccions no s'executen després de prémer la tecla RETORN. L'intèrpret del llenguatge no avaluarà res fins que seleccioneu la comanda **Interpreta** del menú **Edició.** Trobareu pràctic incloure, com a última

línia dels programes, i fora de la definició de cap procediment, la instrucció `amaga.finestra "Edició`. Això farà que la finestra d'Edició es tanqui quan l'intèrpret avaluï el seu contingut, sense que ho hagueu de demanar vosaltres executant la comanda **Tanca** del menú **Finestra**.

Si us semblava un esforç inútil escriure de nou les instruccions cada vegada que les necessitàveu, la finestra d'Edició us compensarà del treball que heu fet fins ara. El menú **Finestra** té dues importants comandes que permeten desar el contingut de la finestra activa en un fitxer, per recuperar intacte tot el treball mes endavant. Es tracta de les comandes **Desa** i **Recupera**, respectivament. Com ja sabeu, també podeu recuperar el contingut d'un fitxer amb la primitiva `recupera`.

Penseu que dins d'un programa podeu escriure instruccions i també comentaris. Els comentaris van sempre precedits pel caràcter punt i coma (`;`), i no afecten de cap forma l'execució de les instruccions. Acostumeu-vos a posar comentaris explicant els vostres programes. Quan els torneu a llegir, temps després d'haver-los creat, us serà més fàcil tornar-los a entendre.

Finalment, per acabar aquesta visió general sobre el treball amb l'editor, cal mencionar dos comandes, molt interessants per treballar amb procediments, que trobareu al menú **Sistema**. En primer lloc, la comanda **Procediments/Variables...**, que permet seleccionar un o més procediment, entre tots els que s'hagin definit al llarg de l'actual sessió de treball, i realitzar diverses operacions amb ells, com ara portar-lo a l'editor. La comanda **Arbre de Procediments...** obre un quadre de diàleg en que el WIN-LOGO pot mostrar la relació de dependència entre

diferents procediments, o indicar si un procediment té crides recursives dintre seu.

Geometria de la tortuga

Ara que coneixeu els procediments podem tractar, amb molta més comoditat, diferents conceptes geomètrics habitualment treballats amb el Logo, i podreu fer dibuixos més ambiciosos que els fets fins ara.

La geometria és la part de la matemàtica basada en la intuïció d'espai. Encara que per tots nosaltres l'espai és una cosa molt concreta, l'estudi tradicional de la geometria, inevitablement, té un caràcter força abstracte. El plantejament geomètric original del Logo, sense referents externs com poden ser els eixos de coordenades cartesianes, dóna un caire totalment subjectiu a l'estudi de la geometria.

Aquest enfocament geomètric, descrivint l'espai sempre en relació a la posició actual de la tortuga, afegit a la dinàmica de treball pròpia de l'ordinador, dóna lloc a un medi d'aprenentatge nou i atractiu.

La metàfora d'aprenentatge fonamental del Logo consisteix en l'activitat d'ensenyar a la tortuga. Ja no és l'alumne el què ha d'aprendre, si no que és a través de la tasca d'instruir la tortuga que aprèn nous conceptes, fent sempre una activitat engrescadora i gairebé lúdica. Anem a ensenyar geometria a la tortuga? Unes senzilles figures seran el pretext, i nous procediments, el mitjà.

Quadrat

És tradicional agafar el quadrat com a la primera figura que s'ha d'ensenyar a dibuixar a la tortuga. L'objectiu és

afegir, a les primitives, un nou procediment capaç de dibuixar un quadrat complet. La nova ordre s'ha de basar en les ja conegudes, o en altres procediments prèviament predefinits. Veiem la primera aproximació. Observareu que algunes línies comencen amb un punt i coma (`;`). Aquestes línies són comentaris, que no afecten l'execució del procediment, i que podeu ignorar o modificar al vostre gust.

```
;;; Primera versió
procediment quadrat
    ;; un costat per línia
    avança 50 gira.dreta 90
    avança 50 gira.dreta 90
    avança 50 gira.dreta 90
    avança 50 gira.dreta 90
fi
```

Una vegada escrit el procediment a la finestra d'Edició, executeu la comanda **Interpreta** del menú **Edició**. Ara el Logo disposa d'una ordre nova, que podeu teclejar dins la finestra de Treball.

```
?quadrat
```

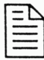
Obedientment, la tortuga dibuixarà un quadrat. Aquesta manera de treballar ja suposa tot un avanç respecte al tedi que representava escriure les instruccions una a una. Podem millorar el procediment, afegint a `quadrat` una entrada. Efectivament, els procediments també poden, com les primitives, tenir entrades. La definició de les entrades d'un procediment té una sintaxi una mica especial. A la mateixa línia en què hi ha la primitiva `procediment` i el nom del procediment, s'han d'escriure els noms de les entrades precedits de dos punts (`:`).



Aquests noms és denominen *paràmetres formals* del procediment. En la pràctica es comporten com variables locals al procediment. Que una variable és *local* vol dir que no és accessible fora del procediment, és com si no existís fora d'ell. Això permet que diferents procediments facin servir els mateixos noms pels seus paràmetres sense que es produeixi cap conflicte entre els noms.

Per accedir, dins les diferents instruccions que hi hagi dintre del procediment, al valor dels diferents paràmetres, s'han d'escriure de la mateixa forma que a la primera línia, amb els dos punts davant. Ara podem definir un procediment que dibuixi un quadrat de qualsevol mida.

```
;;; Segona versió
procediment quadrat :costat
    avança :costat gira.dreta 90
    avança :costat gira.dreta 90
    avança :costat gira.dreta 90
    avança :costat gira.dreta 90
fi
```



En cridar a la nova ordre, no heu d'oblidar d'escriure al seu costat un valor d'entrada adequat. Aquests valors, que poden canviar en cada crida del procediment, s'anomenen *paràmetres actuals*, per diferenciar-los dels paràmetres formals. De totes formes, no cal que recordeu aquesta terminologia si no us ajuda a entendre millor com funcionen les entrades dels procediments. Proveu la nova ordre.

```
?quadrat 50
?quadrat 100
```

Les instruccions anteriors dibuixaran dos quadrats amb mides diferents, actuant com si l'expressió `:costat` del procediment hagués estat substituïda pel valor 50 i 100 en

cada cas. Incorporant una entrada al procediment `quadrat` l'hem dotat d'una gran generalitat, però l'ordre encara està lluny de treballar de forma econòmica. Fixeu-vos que les quatre línies tenen les mateixes instruccions, ja que un quadrat es caracteritza per tenir els quatre costats de la mateixa mida i els quatre angles de 90°.

Per executar repetidament un grup d'instruccions, sense haver de recórrer a la recursivitat, el Logo ofereix l'ordre `repeteix (rep)`. La primera entrada d'aquesta primitiva ha de ser un nombre enter, indicant el nombre de repeticions, i la segona ha de ser la llista d'instruccions que s'ha d'executar repetidament. Les llistes, encara que un element essencial del Logo, són noves per vosaltres.

Una llista és, al Logo, una sèrie de paraules o llistes escrites entre una parella de claudàtors. La definició de llista és, com podeu apreciar, recurrent. L'única regla que heu de recordar, si una llista té dintre seu altres llistes, és que el nombre total de claudàtors oberts ha de ser igual al de claudàtors tancats. Per agafar una mica de confiança a les llistes, teclegeu les instruccions següents dins la finestra de Treball.

```
?mostra [una llista amb 5 elements]
?mostra [[una llista amb] 3 elements]
```

La llista del segon exemple té com a primer element una llista de tres elements. Tornarem a les llistes més endavant, però abans de deixar-les per ara, proveu la primitiva `escriu`. A diferència de `mostra`, ignora els claudàtors exteriors de les llistes en el moment d'escriure-les, i com a novetat, accepta un nombre variable d'entrades.

```
?escriu [una llista amb 5 elements]
?escriu [[una llista amb] 3 elements]
```

Per cridar `escriu` amb més d'una entrada heu de tancar tota la instrucció entre parèntesis.

```
?(escriu "un 2 [i tres entrades])
```

Oblideu, de moment, l'anterior explicació sobre les llistes, i tornem al procediment `quadrat`. Fent servir la primitiva `repeteix`, és suficient escriure les instruccions per dibuixar un costat una sola vegada.

```
;;; Tercera versió
procediment quadrat :costat
  ;; Tota la instrucció en una sola línia
  repeteix 4 [avança :costat gira.dreta 90]
fi
```

Realment, el procediment quadrat ha millorat molt. De totes formes, un procediment no està acabat quan ja funciona i és eficient, sinó quan a més a més és "bonic". L'exigència tradicional del Logo de no dividir les instruccions en línies diferents ens ha obligat a escriure una instrucció molt llarga. Afortunadament, el WIN-LOGO disposa de medis per millorar aquest aspecte; en concret, d'un salt de línia especial anomenat *línia virtual*, que el WIN-LOGO mostra amb el caràcter de cometes franceses obertes («). Gràcies a això, podreu donar un sagnat adequat a les línies, el que fa molt més evident la intenció de les instruccions i la seva estructura lògica. Recordeu que, per inserir un salt de línia virtual, l'única forma de dividir una instrucció en línies diferents, heu de prémer la combinació de tecles MAJÚSCULES+RETORN.

```
;;; Versió definitiva
procediment quadrat :costat
    repeteix 4 [«
        avança :costat«
        gira.dreta 90«
    ]
fi
```

La tortuga ja sap dibuixar quadrats, i de qualsevol mida. Una vegada un nou procediment està definit, i el seu funcionament provat, el podeu considerar com si fos una nova primitiva. Res diferencia procediments i primitives, fora que aquestes sempre són conegudes pel Logo, i els procediments els ha d'aprendre cada vegada que el WIN-LOGO arrenca de nou. Deseu els vostres procediments al disc en fitxers de noms expressius, i recupereu-los amb l'ordre `recupera` quan els necessiteu.

Circumferència

Amb la primitiva `repeteix` disponible és molt fàcil dibuixar circumferències. Només hem de fer que la tortuga avanci un pas i giri un grau tres-centes seixanta vegades seguides.

```
;;; Primera versió
procediment circumferència
    repeteix 360 [«
        avança 1«
        gira.dreta 1«
    ]
fi
```

La forma que té el Logo de definir-dibuixar una circumferència sol mostrar-se com exemple de claredat i intuïció, en oposició a l'equació que en la geometria cartesiana defineix la circumferència:

$$x^2 + y^2 = r^2$$

També podem generalitzar els dibuixos de circumferències, afegint una entrada a la definició del procediment.

```
;;; Versió general
procediment circumferència :passes
    repeteix 360 [«
        avança :passes«
        gira.dreta 1«
    ]
fi
```

D'aquesta forma podreu fer que la tortuga dibuixi circumferències de mides diferents. Proveu les instruccions següents.

```
?circumferència 0.5
?circumferència 1
?circumferència 1.5
```

Queda bé? També és fàcil dibuixar el diàmetre de la circumferència. Coneixent la tradicional fórmula

$$\text{longitud} = 2 \times \pi \times \textit{radi}$$

o el que és igual

$$\text{longitud} = \pi \times \textit{diàmetre}$$

i el valor de π, retornat per la funció `pi`, és fàcil fer una creu dins d'una circumferència. Fixeu-vos que coneixem la longitud de la circumferència —360 passes—, ja que la instrucció `circumferència 1` dóna lloc a repetir 360 vegades la instrucció `avança 1`.

```
?circumferència 1
?gira.dreta 90
?avança 360 / pi
```

```
?recula 360 / pi / 2
?gira.esquerra 90
?avança 360 / pi / 2
?recula 360 / pi
?desapareix
```

Triangle equilàter

Dibuixar un triangle equilàter és fàcil. L'únic que pot causar confusió és el fet que els graus que hem de fer girar a la tortuga són els de l'angle exterior, 120, i no els de l'interior del triangle, que són 60.

```
procediment equilàter :costat
    repeteix 3 [«
        avança :costat«
        gira.dreta 120«
    ]
fi
```

Si volem que la base del triangle sigui horitzontal, cal encarar a la tortuga adequadament abans de començar a dibuixar.

```
?gira.dreta 30
?equilàter 75
```

D'on creieu que surt el valor de 30° que s'ha de girar per encarar correctament la tortuga?

Rectangle

Un rectangle és molt semblant a un quadrat, i la definició d'un procediment per dibuixar-lo no presenta cap dificultat.

```
procediment rectangle :base :alçada
    repeteix 2 [«
        avança :alçada«
        gira.dreta 90«
        avança :base«
        gira.dreta 90«
    ]
fi
```

Fixeu-vos en que `rectangle` té dues entrades, ja que la base i l'alçada dels rectangles són de mides diferents.

Repetició de patrons

Una tècnica espectacular per fer dibuixos consisteix a dibuixar una figura elemental, com ara un quadrat, girar la tortuga un determinat nombre de graus, repetir la figura, i continuar així fins que la tortuga hagi fet la volta completa. Veieu uns quants exemples.

```
?inicia.dibuix
?repeteix 12 [quadrat 50 gira.dreta 30]

?inicia.dibuix
?repeteix 8 [quadrat 50 gira.dreta 45]

?inicia.dibuix
?repeteix 12 [equilàter 75 gira.dreta 30]

?inicia.dibuix
?repeteix 8 [equilàter 75 gira.dreta 45]
```

També s'obtenen dibuixos interesants fent petits desplaçaments abans de cada gir, com mostra el procediment següent.

```
procediment dibuix
  repeteix 6 [«
    avança 20«
    gira.dreta 60«
    quadrat 75«
  ]
  desapareix
fi
```



Podríeu dir quina relació hi ha entre els graus que la tortuga gira entre cada figura, i el nombre de repeticions que cal fer per completar la volta? Experimenteu amb altres figures i girs.

Noves funcions

Fins ara hem presentat procediments que es comportaven com ordres, però també és possible definir noves funcions amb la primitiva `procediment`. Per tal que un procediment sigui una funció ha de contenir la primitiva `retorna` dins la seva definició. Veieu com a exemple la definició de dos simples funcions numèriques.

```
procediment doble :n
  retorna :n * 2
fi
```

```
procediment al.quadrat :n
  retorna :n * :n
fi
```

Una vegada definida, una nova funció pot aparèixer a qualsevol expressió on les seves habilitats siguin necessàries.

```
?mostra doble 3
 6
?mostra al.quadrat 3
 9
```

Retornant a la tècnica per fer dibuixos, mitjançant la repetició de patrons, anteriorment presentada, podem ara veure com una funció pot calcular el nombre de voltes a partir de l'angle que la figura ha de girar entre cada posició.

```
procediment voltes :angle
  retorna quocient 360 :angle
fi
```

Fer servir l'expressió `voltes 45` dins d'una instrucció és equivalent a haver posat directament l'expressió `quocient 360 45`. Ara podeu experimentar amb diferents patrons més facilment, sense que hagueu de calcular el nombre de voltes.

```
?inicia.dibuix
?repeteix voltes 45 [equilàter 75»
gira.dreta 45]
?inicia.dibuix
?repeteix voltes 60 [equilàter 75»
gira.dreta 60]
```



Que passa quan 360 no és múltiple del nombre de graus que feu girar a la tortuga? Per que `voltes` no funciona bé en aquests casos? Com la milloraríeu per que sempre retornés el nombre de voltes necessàries per que la tortuga dibuixi una figura tancada?

Sumari

Els procediments permeten agrupar instruccions, que són executades en el moment de cridar el procediment dins d'una instrucció.

Els procediments, com les primitives, també poden tenir entrades. Els paràmetres són accessibles dins dels procediments a través de variables locals.

La primitiva repeteix permet executar repetidament una llista d'instruccions. El segon paràmetre de `repeteix` ha de ser una llista d'instruccions.

Els procediments poden actuar com ordres o funcions.

Taula 3-1. Primitives tractades

Nom	*Entrades*	*Efecte*
`amaga.finestra`	1	Tanca finestres
`atzar`	1	Retorna un nombre a l'atzar
`escriu`	$1 \Rightarrow \infty$	Escriu els seus paràmetres
`fi`	...	Delimita el fi d'un procediment
`pi`	0	Retorna el valor de la constant π
`procediment`	...	Defineix un procediment
`repeteix`	2	Executa una llista d'instruccions repetidament
`retorna`	1	Dona fi a una funció

Exercicis



Podeu considerar els procediments com petites peces, que poden combinar-se per construir peces majors. Per resoldre els següents exercicis comenceu pels elements individuals de cada problema, un per un, i sense donar-vos compte haureu resolt el problema sencer.

Exercici 3-1. Component diferents quadrats, rectangles, triangles i circumferències, dibuixeu un paisatge senzill o una composició geomètrica. Una casa, un sol, cotxes... Feu servir també línies per dibuixar carrers, arbres, etc. Feu un procediment de nom `paisatge` que centralitzi la realització del dibuix.

Exercici 3-2. Afegiu color al paisatge de l'exercici anterior. Recordeu les primitives `fes.color` i `omple`. Per donar varietat al traç del llapis també us pot ser útil la primitiva `fes.gruix`.

Exercici 3-3. Definiu un procediment, de nom `polígon.regular`, capaç de dibuixar qualsevol polígon regular, a partir del seu nombre de costats. Recordeu, un polígon és regular si tots els seus costats i angles són iguals. Això implica que la suma dels angles exteriors és igual a 360°.

Exercici 3-4. Si no tinguéssim ja un procediment per dibuixar quadrats podríem aprofitar el procediment `rectangle` per definir-lo. En efecte, podem considerar els quadrats com un subconjunt dels rectangles, i cridar dins d'un procediment una ordre definida per nosaltres, igual que fem amb les primitives. Definiu un procediment `quadrat` que per fer el dibuix cridi al procediment `rectangle`.

Exercici 3-5. Dibuixeu els anells olímpics. Feu el dibuix amb els colors de Barcelona 92.

Exercici 3-6. Dibuixeu la façana d'un castell component quadrats i rectangles.

Material complementari



La generalitat i la modularitat són eines que permeten assolir fites aparentment impossibles si només mirem els problemes de lluny i en la seva totalitat. Treballeu amb disciplina i de forma endreçada, enfocant la resolució dels problemes començant pels seus elements més petits.

Exercici 3-7. Feu procediments que generalitzin els dibuixos basats en la repetició de patrons. Poden tenir noms similars a, per exemple, `quadrat.repetit`, i acceptar com entrada els graus que la tortuga ha de girar entre cada figura. Haureu de calcular el nombre de repeticions en base a aquest paràmetre, o fer servir la funció `voltes`, presentada anteriorment.

Exercici 3-8. Aprofitant els procediments següents, feu una composició "floral". Comenceu per definir un procediment de nom `flor` que dibuixi flors de mides diferents i amb un nombre de pètals variable. Trobareu els procediments definits dins del fitxer ARC.LOG —fixeu-vos que un pètal està constituït per dos arcs simètrics—

```
procediment arc :radi :angle
  repeteix part.entera ( pi * :radi * :angle / 180 )[«
      avança 1«
      gira.dreta 180 / ( pi * :radi )«
  ]
fi
```

```
procediment pètal :radi :angle
  repeteix 2 [«
      arc :radi :angle«
      gira.dreta 180 - :angle«
  ]
fi
```

Exercici 3-9. Recupereu els procediments que heu definit per realitzar l'execici 3-1 i modifiqueu el seu funcionament per tal que puguin dibuixar paisatges a escales diferents. Feu que el procediment `paisatge` accepti com a entrada l'escala del dibuix. Per obtenir el mateix dibuix, però amb diferents mides, haureu d'escalar totes les crides a `avança` amb una multiplicació. Per treballar amb més comoditat, podeu fer servir una variable global; per exemple de nombre `escala`.

Exercici 3-10. Per provar diferents combinacions de color, feu que els colors emprats als dibuixos s'agafin d'una sèrie de variables globals. Podeu fer servir noms de variables com ara `cel`, `fusta`, `terra`, `ferro`, etc. Modifiqueu `paisatge` per poder fer els dibuixos amb colors diferents, o fins i tot amb colors a l'atzar (recordeu la primitiva `atzar`).

Exercici 3-11. Si al procediment `circumferència` feu girar a la tortuga un nombre negatiu de graus cap a la dreta, el dibuix es farà al revés, com si féssiu girar a la tortuga cap a l'esquerra. Modifiqueu el procediment per tal que accepti com a entrada un valor, els nombres 1 o -1, a fi de poder dibuixar circumferències cap a l'esquerra o la dreta.

Exercici 3-12. Definiu una funció que, donat el radi d'una circumferència, retorni la seva superfície. A

continuació, teniu com recordatori, la fórmula que heu d'aplicar per calcular el nou valor.

$$\text{superfície} = \pi \times radi^2$$

4

TÈCNIQUES DE TREBALL



El material presentat fins ara ja dóna per realitzar interessants treballs amb l'alumnat. Fins i tot un petit subconjunt del presentat pot ser suficient als nivells educatius més elementals.

És per això que, i una mica en qualitat de resum, el present capítol serà destinat a realitzar una ample i heterogènia reflexió metodològica, tant sobre les pràctiques educatives en què el Logo pugui participar, les formes d'encarar algunes de les dificultats pròpies del llenguatge, com també sobre diverses tècniques que fan el treball més fàcil, efectiu, i de millor qualitat.

Treball a classe

Les orientacions següents us mostraran algunes tècniques, de caire general, per incorporar el Logo a la vostra pràctica educativa.

El paper del professorat

L'entorn de treball més propici per l'aprenentatge del Logo serà el que afavoreixi les activitats d'exploració i aprenentatge. És responsabilitat del professor

proporcionar un entorn de treball en el que els estudiants treballin de forma indagadora, o com s'ha vingut a denominar, heurística.

Aprendre explorant no significa deixar treballar els alumnes únicament segons la seva pròpia iniciativa. El professor ha de proporcionar les eines per descobrir, animar i a vegades, suggerir direccions per a l'exploració. Ha de saber fer un equilibri entre la necessitat de donar als nens l'ajuda necessària perquè progressin, i donar llibertat per que experimentin, facin errors i guanyin confiança en si mateixos.

Aprenentatge heurístic

L'aprenentatge heurístic no té res a veure amb saber quina tecla s'ha de prémer en un moment donat, o quan falta o sobra un espai en blanc dins d'una instrucció. Si bé el Logo suggereix que els estudiants han de prendre el control sobre el seu propi aprenentage, necessiten familiaritzar-se amb les característiques del llenguatge, la definició de procediments, l'ús de l'editor, etc., i és aquí on el professor és més necessari.

Perquè els estudiants puguin explorar amb llibertat, el professor els ha d'haver ensenyat les primitives del llenguatge, com fer servir el teclat o el ratolí, cuidar l'ordinador, utilitzar la impressora, desar i recuperar el treball als disquets, etc.

L'ordre en què el professor ha de presentar els diferents conceptes del llenguatge Logo depèn del seu criteri i de les possibilitats i interessos dels alumnes, i encara que tothom podria coincidir en la forma de començar, els

camins a seguir poden ser molt diferents en funció dels gustos, interessos o capacitats de cada professor.

Cal saber aprofitar els descobriments realitzats pels alumnes per introduir, de forma més raonada, els conceptes que es puguin trobar darrera de l'experiència. El professor també ha de fomentar la capacitat de fer prediccions i provar-les sistemàticament, de treballar metòdicament vers un objectiu definit, dividir els problemes en subproblemes, etc.

El racó del Logo

Crear un entorn d'aprenentatge efectiu necessita d'una organització espacial raonada, i adaptada a cada cas particular.

Amb els nens més petits, és fonamental disposar d'un ordinador a l'aula. Dos o tres alumnes treballen bé en grup, i poden fer servir l'ordinador mentre els altres nens estàn ocupats en diferents activitats. S'ha de tenir cura de controlar les possibles desviacions que el treball en grup pot provocar: nens inhibits o dominants, diferències en funció del sexe, etc.

Temps de treball

Una organització flexible del temps d'ús de l'ordinador sol ser la més adequada. Mentre alguns grups poden assolir el seu objectiu en quinze o vint minuts, pot resultar que un grup força engrescat en una feina necessiti treballar tot un matí sencer.

Per aprofitar al màxim el temps davant de l'ordinador, cal que els diferents grups planifiquin per endavant la seva feina. De totes formes, quan el treball amb l'ordinador

porti a llocs insospitats, s'ha de ser flexible i deixar aparcat el projecte inicial.

Mentre els grups treballen amb l'ordinador el professor pot simplement observar, i intervenir només en cas de necessitat. Sobre aquest aspecte es poden trobar, de totes formes, opinions molt diferents. Quan i com ha d'intervenir el professor? Ha d'intervenir? Normalment, ja seran els propis alumnes els que demanin ajuda quan estiguin en un cul-de-sac.

La feina feta

Pot ser convenient que cada alumne disposi d'una carpeta relacionada amb el treball amb el Logo. S'hi poden desar les descripcions de les primitives, els dibuixos trets per la impressora, els procediment definits, etc. Apart de la feina personal i del grup, el professor pot considerar útil repartir fotocòpies dels millors dibuixos realitzats, exemples de Logo encertats o triats especialment, etc.

El taulell d'anuncis de la classe pot ser un bon lloc per penjar dibuixos fets amb el Logo. També es poden fer composicions amb diferents dibuixos, o fins i tot preparar un cartell amb diferents dibuixos creats de forma cooperativa, tenint el conjunt final en perspectiva, etc.

L'aula d'ordinadors

Quan arriba el moment d'anar a l'aula d'ordinadors per fer Logo? S'ha de fer "classe" de Logo? És el Logo una nova assignatura? Com podem relacionar el Logo amb les assignatures tradicionals? Aquestes i altres preguntes es plantegen tard o d'hora quan es treballa amb el Logo.

A partir d'una determinada edat, portar els alumnes a l'aula d'ordinadors es converteix en una necessitat i una activitat de gran profit. El ritme de treball pot accelerar-se enormement, i la interacció entre els diferents grups fer-se més profitosa i engrescadora. De totes formes, treballar amb molts ordinadors alhora en lloc de tan sols amb un no és massa diferent, i les mateixes tècniques que es poden aplicar al racó del Logo poden ser útils a l'aula d'ordinadors. El que si és nou són les preguntes amb que començàvem aquest apartat. Podeu trobar-ne resposta?

Orientació de la tortuga

Així com les ordres de desplaçament de la tortuga no plantegen cap dificultat especial d'aprenentatge, no s´han de menysprear les dificultats que pugui presentar el gir i l'orientació de la tortuga. Els següents paràgrafs són recordatori de les dificultats, i les possibles solucions, sempre opinables, que aquest tema pot tenir.

Gir absolut

La dreta i l'esquerra de la tortuga no sempre són la dreta i l'esquerra de la pantalla de l'ordinador. Si tombeu una pantalla ho comprovareu immediatament. Tampoc la nostra dreta i esquerra coincideixen amb les de la tortuga. Per exemple, quan la tortuga mira cap avall, la seva dreta és la nostra esquerra.

Potser us agradaria més parlar de voltes senceres i de fragments de volta. Si decidim que el nombre 1 representi una volta sencera cap a la dreta, el -1 una volta sencera cap a l'esquerra, i el 0 representi el gir nul, podem fer girar a la tortuga sense pensar en dreta i esquerra. A

continuació teniu la definició de l'ordre `gir`, i una versió de `quadrat` que la fa servir.

```
;;; Gir absolut
procediment gira :fracció
  gira.dreta 360 * :fracció
fi

;;; Quadrat dibuixat girant un quart
procediment quadrat :costat
  repeteix 4 [«
    avança :costat«
    gira 1/4«
  ]
fi
```

Coordenades

No pot dir-se que la divisió de la circumferència en 360° sigui un concepte "natural". Ara bé, si relacionem els graus amb les coordenades geogràfiques, potser anirem més "orientats". L'orientació geogràfica, amb els tradicionals punt de l'horitzó —nord, sud, est i oest—, fa que puguem guiar a la tortuga com si tinguéssim una brúixola.

La primitiva `orienta't` permet orientar la tortuga de forma absoluta, no relativa a la seva posició actual. Per conèixer l'orientació de la tortuga podeu fer servir, en qualsevol moment, la funció `orientació`. Amb aquestes primitives podem definir un procediment que dibuixi un quadrat horitzontal, independentment de l'orientació de la tortuga en el moment de cridar al procediment. Fixeu-vos que el nom del procediment s'aprofita també com variable per emmagatzemar l'orientació original de la tortuga, que és restaurada abans d'acabar.

```
;;; Quadrat sempre en la mateixa posició
procediment quadrat.horitzontal :costat
  ;; Guardem l'orientació original
  posa.a "quadrat.horitzontal orientació
  ;; Dibuixem el quadrat
  orienta't 0          ;nord
  avança :costat
  orienta't 90         ;est
  avança :costat
  orienta't 180        ;sud
  avança :costat
  orienta't 270        ;oest
  avança :costat
  ;; Restaurem l'orientació original
  orienta't :quadrat.horitzontal
fi
```

Angle exterior

A l'hora de dibuixar figures tancades, l'angle que s'ha de girar en els diferents vèrtex és l'angle exterior, no l'interior. Això pot resultar confús, especialment si ja s'han estudiat les figures geomètriques de la forma tradicional. Per continuar mantenint els conceptes tradicionals, podeu fer servir la funció `exterior`, presentada a continuació. També es presenta un nou procediment per dibuixar triangles equilàters. Compareu la seva definició amb la donada al capítol anterior.

```
;;; Converteix en exterior un
;;; angle interior
procediment exterior :angle
  retorna 180 - :angle
fi
```

```
;;; Triangle definit amb angles interiors
procediment equilàter :costat
   repeteix 3 [«
        avança :costat«
        gira.dreta exterior 60«
    ]
fi
```

Aprenentatge per atzar

Potser algú s'escandalitzarà de l'enunciat que dóna títol a aquest apartat. És possible aprendre per "casualitat"? Els defensors del pensament lateral, entès com oposat a l'analític, propugnen que és possible trobar solucions sense cercar-les, o fins i tot triant un camí equivocat, o quan es buscaven altres coses...

Errors

Treballant amb el Logo, moltes vegades les coses no funcionen tal com s'esperava. A vegades simplement hem comès un error d'escriptura, hem canviat una lletra per una altra, hem oblidat posar espai on feia falta, etc. A vegades pot ser que estiguem equivocats realment, i que els nostres programes tinguin veritables errors. Resoldre els errors de programació és una tasca molt educativa, en què hem d'aprendre a qüestionar la nostra concepció de les coses, i dubtar de totes les conviccions suposadament encertades que tinguem. La desdramatització de l'error és, potser, la més radical aportació del Logo a la didàctica moderna, i ens ensenya que tots fem errors. El que és important no és cometre errors, sinó saber com resoldre'ls.

Encara que parlar a fons de les tècniques per enfrontar-se als problemes de programació està fora de l'abast d'aquest curs, no podem deixar de parlar de les importants eines de què el WIN-LOGO disposa en aquesta àrea, especialment les disponibles a l'entorn de Traçat.

En activar la finestra de Traçat, executant la comanda **Traçat** del menú **Entorns**, apareix a la barra de menús el menú **Traçat**. Les seves opcions permeten executar qualsevol instrucció pas a pas, fer-ho de forma animada, establir punts d'aturada... Totes les instruccions que s'executen amb la finestra de Traçat oberta s'escriuen dins d'aquesta finestra, com també s'escriuen els valors d'entrada dels procediments. Aquest entorn de treball pot ser d'una ajuda inestimable, no tan sols per trobar els possibles errors dels nostres programes, sinó també per entendre millor el seu funcionament.

Pensament lateral

No sembla necessari defensar com el treball amb ordinadors pot potenciar el pensament analític. Hi ha qui diu que són els ordenadors els que ens "ordenen" a nosaltres... El que a molts estranyaria és pensar que els ordenadors també poden potenciar el pensament lateral.

A vegades, el que pot semblar un error dins d'un procediment, no ho és en realitat. Potser el procediment no fa el que nosaltres preteníem inicialment, però el que fa no està malament. A vegades, està molt bé. Seymour Papert, el pare del Logo, menciona al seu llibre *Desafío a la mente* com un pètal de flor mal dibuixat pot passar a ser les ales d'un ocell. Treballant amb Logo, mai sabem que trobarem.

Hi ha problemes que es resisteixen endimoniadament, fins que ens fan desistir d'intentar resoldre'ls. Més endavant, quan ja potser el problema està totalment oblidat, cridem: *Eureka!* El que estem fent ara resol el problema que ens va ocupar abans. Com pot ser que una solució, que tant es va resistir, arribi per si mateixa? Hi ha qui ha trobat resposta a aquesta pregunta. No és més que una manifestació, diuen, de la nostra capacitat de tenir pensaments laterals.

Ús de la impressora

Disposar d'una còpia impressa dels dibuixos fets per la tortuga és una cosa que a tothom fa gràcia, i especialment als nens i nenes. Tan aviat com domineu l'ús de la impressora li podreu treure un gran profit, i encara que no és fàcil, l'esforç us serà recompensat pels resultats.

Configuració

El WIN-LOGO disposa de tres comandes, situades al menú **Utilitats**, per gestionar i controlar la impressora. La informació i complexitat associades a aquestes comandes és força gran, i aquí només farem esment del conceptes fonamentals. Després de la lectura d'aquest text, l'experimentació serà la millor eina que trobareu per treure profit de la vostra impressora.

La comanda **Configura impressora...** obre un quadre de diàleg en què podeu informar al WIN-LOGO del model d'impressora que teniu i el port de l'ordinador en què està connectada. Aquesta informació és necessària, i l'heu de proporcionar vosaltres.

El quadre de diàleg que es visualitzarà en seleccionar la comanda **Format impressió text...** us permet definir la mida de paper, els marges que voleu deixar, etc. Penseu que el contingut de les finestres de Treball, Text i Edició pot ser imprès, i que és en aquesta finestra on es defineixen els paràmetres amb què es realitzarà la seva impressió.

El quadre de diàleg que obre la comanda **Format impressió gràfic...** disposa de diverses opcions per controlar la impressió de la finestra de Gràfics. Una opció d'aquest quadre de diàleg força interessant és **Posició paper**, amb la qual podeu fer que els dibuixos s'imprimeixin apaïsats. Els valors relacionats amb els **Marges** i la **Mida del gràfic** són, en teoria, els òptims per a la mida inicial de la finestra de Gràfics, però si modifiqueu les dimensions d'aquesta finestra, haureu de canviar també aquests valors. Penseu que la relació entre l'amplada i l'alçada del dibuix depèn de la mida de la finestra de Gràfics, pel que no podeu triar valors arbitraris. Amb l'ajuda de la primitiva `mida.finestra`, que retorna les mides d'una finestra qualsevol, podeu calcular els valors en centímetres per fer una impressió correcta. Les instruccions següents us mostren com fer-ho, encara que els valors són els d'un monitor VGA. Les constants que es resten a l'amplada i llargada estan en relació a la mida del marc de la finestra, i el valor 0.03 és arbitrari —i modificable—, i determina una ocupació de la fulla en uns dos terços de la seva mida.

```
?mostra mida.finestra "Gràfics
 [640 448]
?mostra 640-24 * 0.03   ;amplada
 18.48
```

```
?mostra 448-64 * 0.03   ;llargada
 11.52
```

L'ordre executiva per iniciar la impressió d'una finestra es troba al menú **Finestra**. Quan trieu la comanda **Imprimeix**, el contingut de la finestra activa serà enviat a la impressora.

Totes les dades relacionades amb l'impressió agafen els seus valors per defecte de tres línies que figuren al fitxer de configuració del WIN-LOGO, de nom WLOGO.INI. Si teniu coratge per editar aquest fitxer, i adaptar els paràmetres d'impressió al vostre gust, consulteu el *Manual de Referència* del WIN-LOGO per saber com fer-ho.

Tècniques especials

Una tècnica vàlida per tenir constància del treball dels vostres alumnes, però també útil per vosaltres ara que esteu aprenent, consisteix a capturar tot allò que s'escrigui a les finestres de Treball i de Text dins d'un fitxer de text. Després d'una sessió de treball podeu consultar el contingut del fitxer i repassar tot el que heu fet, on heu comès errors, quin ha estat el resultat de certa operació, etc. Per iniciar el registre d'una sessió de treball dins d'un fitxer, per exemple, de nom CAPTURA.TXT, heu d'executar la instrucció següent.

```
?connecta "captura.txt
```

L'ordre `connecta` fa que el Logo enviï eco a un fitxer. Per interrompre aquesta tasca, heu de cridar a la primitiva `desconnecta`.

```
?desconnecta
```

Més endavant podeu llegir el contingut del fitxer, per exemple amb l'editor del WIN-LOGO.

Una alternativa a la captura del text en un fitxer és enviar-lo a la impressora. Per fer-ho, heu de cridar a `connecta` amb la paraula "màgica" `prn` —abreviatura de la paraula anglesa *printer*— per paràmetre.

```
?connecta "prn
```

Si la vostra impressora està connectada i té paper, començarà a recollir tot els teclegeu a la finestra de Treball o el que aparegui a la finestra de Treball.

Problemes

Les impressores es caracteritzen per ser unes màquines de comportament una mica capritxos. Els problemes que una impressora pot provocar són molt nombrosos, i a més, cada impressora té la seva personalitat i la seva pròpia col·lecció de problemes. Amb una mica de sort podreu arribar a conèixer la vostra impressora, però mai de la vida penseu que podreu fer que totes les impressores es rendeixen als vostres desitjos. No us sembla que aquesta descripció de la relació amb les impressores recorda una descripció del matrimoni?

Estil de programació



Fer servir una estètica correcta per l'escriptura del codi és, desgraciadament, un aspecte de la programació a vegades menystingut o oblidat. La seva importància, però, és cabdal. Sobre aquest tema no poden dictar-se regles d'aplicació universal, ja que no afecten a l'eficàcia dels programes, i depenen en gran mida dels gustos de cadascú.

Les normes següents són les emprades en aquest manual, i s'aconsella que les feu servir també en els vostres propis programes, a falta d'altres de personals que decidiu adoptar.

Noms

Triar el nom de procediments i variables és un privilegi al qual no podem renunciar. Del criteri que apliquem en escollir els noms se'n poden derivar programes llegibles o modernes versions de l'escriptura jeroglífica. El nom de les primitives, tal com ha estat dissenyat al WIN-LOGO català, pot servir d'inspiració per decidir l'estil per anomenar els procediments i les variables. Algunes normes bàsiques podrien ser les següents: triar noms significatius, descriptius; no fer servir abreviatures; escollir, per les ordres, verbs en forma imperativa; en els noms formats per dos o més parts, separarles amb un punt i finalment, no oblidar que un codi poc clar acaba portant a un codi que no funciona.

El sagnat

En l'idioma anglès es fa servir l'expressió *pretty-printing* per fer referència a tots aquells aspectes de l'escriptura dels programes que determinen, amb l'objectiu d'afavorir la seva lectura i comprensió, la forma en què es distribueixen els textos al llarg del paper (o pantalla). El sagnat, principal arma del *pretty-printing,* consisteix a entrar de forma diversa les línies de codi per posar de relleu l'estructura del programa. Inserir caràcters de tabulació és la forma més simple d'entrar les línies, i és la tècnica que s'ha aplicat a l'exemple següent.

```
procediment deu.salutacions
      repeteix 10 [«
            escriu "Hola!«
            escriu [Bon dia]«
      ]
fi
```

Cercant la màxima simplicitat sintàctica, el Logo no exigeix l'ús de cap caràcter de puntuació per separar les diferents instruccions, però fa recaure sobre el fi de línia una responsabilitat massa gran. Sense anar més lluny, no permet dividir l'escriptura de les llistes entre diverses línies. Si bé el llenguatge Logo permet disposar lliurement dels espais en blanc, imposa grans restriccions pel que fa als finals de les línies. El Logo no permet "trencar" les instruccions amb el caràcter de fi de línia, al que considera com a separador de diferents instruccions.

Les versions més antigues de Logo no permetien trencar les línies de cap forma, cosa que obligava a escriure complexes instruccions en una única línia, provocant una desagradable obscuritat del codi. Afortunadament, WIN-LOGO incorpora el concepte de línia virtual, que s'insereix en prémer les tecles MAJÚSCULES+RETORN. L'efecte visual és el mateix que provoca el caràcter de fi de línia, fora de la visualització del caràcter de fi de línia virtual («), però sense el significat afegit de representar un final d'instrucció. És, doncs, altament recomanable, per tal d'augmentar la llegibilitat dels programes, fer un ús generós del final de línia virtual.

Una norma de bon gust

Cal escriure tan sols una instrucció per línia, fora de casos justificats. Quins són aquests casos? Per exemple, podem deixar juntes en una línia una ordre d'avançar i una de

girar, per emfasitzar la unitat que formen el desplaçament i el gir. Quins altres? Es qüestió de bon gust!

Comentaris

Si el codi Logo s'escriu per tal que el "llegeixi" l'intèrpret del llenguatge, els comentaris van dirigits a les persones, i no afecten de cap manera a l'execució dels programes. Cap de les regles sintàctiques que regeixen el llenguatge de programació han de seguir-se dins dels comentaris. Aquests han d'estar escrits de forma entenedora, clar i en català.

L'ús generós de comentaris facilita la comprensió i manteniment dels programes. En primer lloc, però, cal intentar que el codi es documenti a si mateix. No té massa sentit programar un codi incomprensible, que necessita molts comentaris per poder ser llegit. Deixant de banda aquesta recomanació, sempre que calgui aclarir el comportament del codi s'ha d'afegir un comentari, precedit del caràcter de punt i coma (`;`).

Sumari

La metodologia de treball proposada pel Logo es basa en l'exploració i l'experimentació.

El paper del professor, en treballar amb el Logo, no consisteix només a trametre coneixements, sinó en ajudar a descobrir-los.

Cometre errors és natural, i saber-los resoldre, l'objectiu final de qualsevol aprenentatge.

Escriure amb un llenguatge de programació té una dimensió estètica que no s'ha de menysprear.

Taula 4-1. Primitives tractades

Nom	*Entrades*	*Efecte*
`orienta't`	1	Orienta la tortuga
`orientació`	0	Retorna l'orientació de la tortuga
`connecta`	1	Envia eco de tot el text a un fitxer
`desconnecta`	0	Tanca el fitxer d'eco
`mida.finestra`	1	Retorna les mides d'una finestra

Exercicis



Parleu als vostres alumnes del Logo. Segons la seva edat realitzeu algun joc de vivenciació, o porteu-los a l'aula d'ordinadors de la vostra escola. Trieu l'activitat que més us convingui. Després d'aquesta experiència intercanvieu opinions sobre ella amb altres companys i companyes.

Si realitzar aquesta activitat ara altera la vostra planificació actual, trieu el moment adequat més endavant. Aquest capítol no té més exercicis que aquest. És prou difícil com perquè hi dediqueu tota la vostra energia!

5

LÒGICA

Fins ara hem presentat exemples senzills de Logo que incorporaven en la seva estructura —entesa com organització general de les instruccions—, la modularitat que ofereixen els procediments, i un notable grau de generalitat, assolit a través dels paràmetres dels mateixos procediments. Pel que fa a la forma en què les instruccions se succeeixen les unes a les altres, les categories presentades encaixen dins les formes d'execució seqüencial i d'execució repetitiva.

El pas següent consisteix a dotar als programes d'un grau més d'intel·ligència, fent-los capaços de prendre decisions. I per prendre decisions, els programes han de poder opinar, i distingir si una expressió és certa o falsa. Amb aquesta capacitat, els programes poden incloure l'anomenada execució condicional, en la qual determinats fragments del programa són avaluades o no, en funció de la veritat o la falsedat de determinades condicions.

La presentació dels valors lògics obre la porta a característiques noves del llenguatge Logo, i permet mirar amb ulls nous d'altres que s'han presentat abans informalment. Això farà que aquest capítol inclogui un material heterogeni, però sempre relacionat amb la

possibilitat del llenguatge de fer preguntes i de respondre-les.

Valors lògics

Abans de poder actuar en funció de la veritat o la falsedat d'una expressió, el Logo necessita representar els valors lògics d'alguna forma. Els valors lògics són dos: ver i fals, i entre els llenguatges de programació podem trobar diferents formes de representar-los. Alguns ho fan amb dos nombres, tradicionalment agafant el 0 pel valor de falsedat i el 1 pel valor de veritat; altres llenguatges incorporen un tipus especial de valor, diferent dels valors numèrics o de qualsevol altra mena; el Logo representa els valors lògics amb dues paraules de caire especial.

Constants lògiques

El Logo fa servir les paraules `ver` i `fals` per representar els dos possibles valors lògics. No és gaire habitual incorporar aquestes constants directament als programes, i és més comú trobar expressions que les produeixen. La raó d'això és molt simple. La funció dels valors lògics és permetre fer preguntes, i allà on apareixin les constants `ver` o `fals` ja està tot contestat.

Operadors relacionals

Els operadors relacionals comparen dos valors i retornen un valor lògic. L'operador més versàtil és l'operador d'igualtat (=), amb el qual es poden comparar nombres, paraules i llistes. Per veure com funciona, anem a demanar al Logo que ens mostri el resultat d'avaluar algunes expressions lògiques, és a dir, expressions que són veritat o mentida.

```
?mostra 2+2 = 4
 ver
?mostra 2+2 = 5
 fals
```

Com heu vist, el Logo no tan sols ha après a fer sumes, sinó que ja sap que dos més dos no són cinc.

L'operador d'igualtat també serveix per comparar paraules o llistes.

```
?mostra "Joan = "Joan
 ver
?mostra "Joan = "Lluís
 fals
?mostra [Joan i Maria] = [Joan i Josep]
 fals
?mostra [Joan i Maria] = [Joan i Maria]
 ver
```



Quan l'operador d'igualtat es fa servir per comparar paraules, fa distinció entre les lletres majúscules i minúscules. No està massa clar si això és una característica del Logo o un defecte del WIN-LOGO. Haureu d'anar amb cura, especialment a l'hora d'escriure els valors lògics. Els exemples següents us mostren perquè.

```
?si "VER [escriu "ver][escriu "fals]
 ver
?mostra 1<2 = "VER
 fals
?mostra (2+2 = 4) = "ver
 ver
```

Per no tenir problemes convé, doncs, escriure les paraules `ver` i `fals` sempre amb minúscules. Encara millor, es poden cercar formes d'evitar l'aparició de les constants lògiques dins dels programes. Per exemple, les

instruccions següents mostren dues expressions per complet equivalents. La segona no incorpora la paraula `ver` i a més, és més eficient que la primera, ja que evita una comparació inútil.

```
?mostra (2+2 = 4) = "ver
 ver
?mostra (2+2 = 4)
 ver
```

Si la paraula a eliminar és `fals`, podem fer servir la funció `no`, que retorna el valor lògic contrari al que rep com a paràmetre. Podeu imaginar a aquesta primitiva com un commutador que sempre inverteix el valor de la seva entrada.

```
?mostra (2+2 = 4) = "fals
 fals
?mostra no (2+2 = 4)
 fals
```

Els operadors 'major que' (>) i 'menor que' (<) tan sols operen amb nombres. No poden comparar, doncs, ni paraules ni llistes.

```
?mostra 1 < 2
 ver
?mostra 1 > 2
 fals
```

El Logo no disposa, a diferència d'altres llenguatges, dels operadors 'major o igual que' i 'menor o igual que'. Per assolir el mateix resultat s'han de combinar els operadors 'major que' o 'menor que' amb la funció `no`. Considereu que les comparacions 'major o igual que' i 'no menor' són equivalents.

```
?mostra no 1 > 2        ; equival a <=
 ver
```

```
?mostra no 1 < 2          ; equival a >=
 fals
```

Execució condicional

El Logo fa preguntes a través de la primitiva `si`. Aquesta versàtil primitiva segons el context pot actuar com a ordre i com a funció, i és la clau per implementar l'execució condicional.

Condició simple

En la seva forma més simple, `si` espera dos paràmetres. El primer ha de ser una expressió lògica, que per resultat ha de donar els valors `ver` o `fals`, i el segon una llista amb les instruccions que s'hauran d'executar en el cas que el primer paràmetre sigui veritat. Fixeu-vos en les instruccions següents. La primera no té cap efecte, perquè l'expressió `"gat = "llebre` té el valor lògic `fals`, i la llista d'instruccions no és executada. Per contra l'expressió `2+2 = 4` és veritat, el que fa que la primitiva `escriu` escrigui `Ja ho sabia!` a la finestra de Text.

```
?si "gat = "llebre [escriu "Impossible!]

?si 2+2 = 4 [escriu [Ja ho sabia!]]
 Ja ho sabia!
```

Condició doble

Opcionalment, `si` pot rebre una segona llista d'instruccions, que serà executada en el cas que l'expressió lògica sigui mentida. D'aquesta forma, `si` serveix per triar entre dues opcions. La llista que no és executada està en la mateixa situació que es trobaria si no estiguÉs escrita. És com si l'intèrpret del llenguatge anés

llegint el programa, saltant-se determinats bocins durant la seva lectura. A l'exemple següent, la segona llista d'instruccions no És executada mai.

```
?si 3 > 2 [escriu "major] [escriu "menor]
 major
```

Si voleu marejar a la tortuga, repetiu diverses vegades la instrucció següent.

```
?si atzar 2 = 0 [gira.dreta 10 avança 10]»
[gira.esquerra 10 avança 10]
```

El següent procediment li diu al Logo, "Saps que 2 més dos són ...?" Canvieu els anteriors punts suspensius per qualsevol nombre, i proveu d'enganyar el Logo.

```
;;; Saps sumar dos i dos?
procediment saps.sumar? :resultat
  si :resultat = 4 [escriu [Ja ho sabia]]«
                   [escriu "Mentida]]
fi

?saps.sumar 3
 Mentida
?saps.sumar 4
 Ja ho sabia
```

Funcions lògiques

El Logo coneix moltes funcions lògiques, que ja anireu aprenent. També es poden definir funcions lògiques noves, és dir, que poden retornar la paraula `ver` o la paraula `fals`, i res més. Veieu-ne dues de molt senzilles.

```
procediment és.parell :n
  retorna residu :n 2 = 0
fi
```

```
procediment és.senar :n
  retorna no (residu :n 2 = 0)
fi
```

Un cop definida una funció pot aparèixer dins qualsevol expressió.

```
?mostra és.parell 124
 ver
?mostra és.parell 7
 fals
?mostra és.senar 124
 fals
?si és.senar 124 [escriu "SI] [escriu "NO]
 NO
```

La funció `si`

La primitiva `si` també pot actuar com funció. El Logo distingeix els dos usos de la primitiva segons el context.

```
?escriu si atzar 2 = 0 [SI] [NO]
 SI
?gira.dreta si atzar 2 = 0 [10] [-10]
```

Hi ha una circumstància en què la funció `si` és especialment útil, i és quan es combina amb la primitiva `retorna`. Això permet definir funcions, que han de retornar un valor a triar entre dues opcions, amb molta facilitat. Una construcció com la del procediment següent, amb dues vegades la primitiva `retorna` dins seu,

```
procediment parell.o.senar :nombre
  si residu :nombre 2 = 0 [«
    retorna "PARELL ][«
    retorna "SENAR]
fi
```

pot transformar-se en la següent.

```
procediment parell.o.senar :nombre
  retorna si residu :nombre 2 = 0 [«
    "PARELL ][«
    "SENAR]
fi
```

No deixeu de provar qualsevol de les dues versions, amb diferents nombres per paràmetre.

```
?escriu parell.o.senar 33
 SENAR
```

Operacions lògiques

Encara que en un sentit ampli, qualsevol funció que retorni un valor lògic és una funció lògica, aquí considerarem únicament les funcions que permeten combinar diferents valors lògics i en produeixen un de nou. Aquestes funcions són tancades respecte a les seves entrades, cosa que vol dir que els tipus dels seu paràmetres són iguals al del valor que retornen.

Abans ja ha estat presentada la funció no, que accepta un únic paràmetre. Sempre retorna el valor lògic oposat al que rep com paràmetre.

```
?mostra no "ver
 fals
?mostra no "fals
 ver
```

La funció `veres.totes` accepta un nombre indefinit de paràmetres a partir d'un mínim de dos. Retorna `ver` només en el cas que tots els seus paràmetres siguin veritat, i fals en la resta de casos.

```
?mostra veres.totes "ver "ver
 ver
```

```
?mostra (veres.totes "ver "fals)
 fals
```

La funció `vera.alguna` accepta un nombre indefinit de paràmetres a partir d'un mínim de dos. Retorna `ver` si qualsevol dels seu paràmetres és veritat, i fals en la resta de casos.

```
?mostra vera.alguna "ver "fals
 ver
?mostra vera.alguna "fals "fals
 fals
```

Podem combinar diferents funcions lògiques, i naturalment, fer que com a paràmetre rebin expressions lògiques complexes, i no tan sols constants com en els exemples anteriors.

Objectes Logo

Una característica de la família de llenguatges a la que pertany el Logo, és la possibilitat de fer programes amb capacitat d'autoanàlisi, i fins i tot d'automodificació.



El Logo posseeix tres tipus de valors diferenciats, amb operacions internes particulars. Podem sumar nombres, però no un nombre i una paraula. El conjunt d'aquests tres tipus de valors és el que anomenem *objectes Logo*. Entenem per objecte Logo tot tipus de valor que el llenguatge pot manipular.

Tipus d'objectes

Els diferents objectes Logo estan relacionats entre si per una relació de jerarquia. Això vol dir, que si bé tots els valors són objectes, les paraules no són llistes, però els

nombres sí que són paraules, a més de ser nombres. La figura següent mostra aquesta relació.

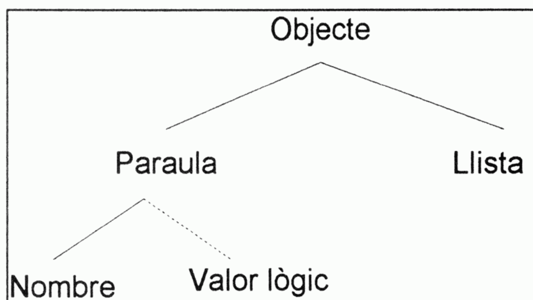


Figura 5-1. Jerarquia de tipus al Logo.

Disposem de diverses funcions lògiques per determinar si un objecte donat pertany o no a un tipus determinat. Els exemples següents us les presentem alhora que veieu el seu funcionament.

```
?mostra és.paraula "Hola
 ver
?mostra és.nombre "Manel
 fals
?mostra és.llista [Hola Manel]
 ver
?mostra veres.totes és.paraula 11»
és.nombre 11
 ver
```

Valors lògics

Encara que els valors lògics no són un tipus de ple dret, raó per la qual en la figura anterior apareixen units a l'esquema amb una línia puntejada, conceptualment pot ser convenient o còmode considerar-los com un tipus

especial. Fins i tot podem definir una senzilla funció per completar les primitives que hi ha en aquest tema.

```
procediment és.lògic :objecte
  si :objecte = "ver  [retorna "ver]
  si :objecte = "fals [retorna "ver]
  retorna "fals
fi
```

Objectes buits

La paraula buida i la llista buida representen el valor nul, un concepte equivalent, en un altre context, al de conjunt buit. Més endavant aprofondirem en el treball amb llistes i paraules. De moment és suficient que sapigueu que dos claudàtors aïllats representen la llista buida, i unes cometes seguides d'un espai a la paraula buida. També està disponible la funció `és.buida`, que permet determinar si un objecte és buit. Teclegeu els exemples següents i fixeu-vos en el darrer: la llista buida i la paraula buida són objectes diferents.

```
?mostra és.buida [ ]
 ver
?mostra és.buida "
 ver
?mostra és.buida "Hola
 fals
?mostra és.buida [Hola, bon dia!]
 fals
?mostra [ ] = "
 fals
```

Sumari

Taula 5-1. Primitives tractades

Nom	*Entrades*	*Efecte*
`és.buida`	1	Determina si un objectes és buit
`és.llista`	1	Determina si un objecte és una llista
`és.nombre`	1	Determina si un objecte és un nombre
`és.paraula`	1	Determina si un objecte és una paraula
`no`	1	Operador lògic "no"
`si`	2 o 3	Permet escollir entre dues opcions
`vera.alguna`	$2 \Rightarrow \infty$	Operador lògic "o"
`veres.totes`	$2 \Rightarrow \infty$	Operador lògic "i"

La primitiva `si` permet seleccionar entre dues opcions.

Les primitives `no`, `veres.totes` i `vera.alguna` produeixen un valor lògic en base a altres valors lògics.

És possible determinar el tipus d'un objecte Logo.

La paraula buida i la llista buida representen el valor nul.

Taula 5-2. Operadors lògics

Caràcter	*Operació*
=	igualtat
<	major
>	menor

Les paraules `ver` i `fals` representen les dues constants lògiques, i manifesten la condició de veritat o falsedat d'una expressió lògica.

Els operadors relacionals comparen dos valors, i retornen un valor lògic.

Exercicis



A mida que avancem, i aneu coneixent més aspectes del llenguatge Logo, és més important que us sapigueu plantejar problemes pel vostre compte. Els exercicis següents us han de servir per tenir un model de la mena de problemes que esteu en condició de plantejar-vos.

Exercici 5-1. Definiu una funció de nom `absolut`, que esperi un nombre com a paràmetre, i retorni el valor absolut d'aquest nombre.

Exercici 5-2. Feu un procediment que accepti tres paràmetres numèrics i escrigui a la finestra de text la seva suma i la seva mitjana.

Exercici 5-3. Definiu una funció de nom `signe`, que esperi un nombre com paràmetre, i retorni `-1` si el nombre és negatiu, `1` si és positiu, i `0` si és igual a zero.

Exercici 5-4. Definiu una funció de nom `són.diferents`, que esperi dos paràmetres, i retorni `ver` si són diferents, i `fals` en cas contrari.

Exercici 5-5. Definiu una funció de nom `edat.militar`, que esperi un nombre com paràmetre, i determini si és major o igual de 18 i menor o igual de 20.

Exercici 5-6. Definiu una funció que rebi com a paràmetre qualsevol objecte Logo, i retorni la llista `[paraula buida]` si es tracta d'una paraula buida, i `[llista buida]`si es tracta d'una llista buida. En un altre cas feu que retorni la llista `[no buida]`.

Material complementari



Els exercicis següents no són massa difícils. Intenteu fer-los, o agafeu-los de model per imaginar altres exercicis del vostre gust.

Exercici 5-7. Definiu una funció que determini si un any és de traspàs, segons la descripció següent. Un consell: feu servir la primitiva `residu`.

«Un any és de traspàs si és divisible entre 4, fora de si és divisible entre 100, excepte si és divisible entre 400.»

Exercici 5-8. Tenint en compte que l'expressió en llenguatge Logo `atzar 2 = 0` és veritat en el 50 per cent dels casos, l'expressió `atzar 3 = 0` ho és en el 33 per cent dels casos, i així successivament, milloreu el procediment `passeig` que a continuació us recordem, per ier el passeig de la tortuga més versemblant. Podeu fer que, amb una probabilitat del 90 per cent, la tortuga eviti anar a la part dreta de la pantalla? Feu que la tortuga torni al centre cada cert temps.

```
procediment passeig
   avança atzar 10
   gira.dreta atzar 20
   passeig
fi
```

Exercici 5-9. Què fa el següent procediment?

```
procediment misteri :n :m
  posa.a "misteri 1
  repeteix :m [«
    posa.a "misteri :misteri * :n«
  ]
  retorna :misteri
fi
```

Veieu com treballa:

```
?mostra misteri 2 3
 8
?mostra misteri 3 2
 9
```

Exercici 5-10. Definiu una funció que retorni un nombre aleatori entre un mínim i un màxim donats, ambdós inclosos.

Exercici 5-11. Definiu una funció que determini si el seu paràmetre és una paraula, pero no és un nombre.

Exercici 5-12. Definiu una funció lògica que implementi l'operador lògic "o exclusiu".

6

GRÀFICS

El micromón Logo per excel·lència és el dels gràfics. Ja coneixeu el món de la tortuga, on l'exploració de la geometria es realitza de forma experimental. Aquest capítol presenta tres aspectes fonamentals del món gràfic del Logo, que esperem que siguin suficients per tenir una visió àmplia de les possibilitats del llenguatge en aquest camp.

Polígons

Al tercer capítol, en el qual hem presentat els procediments, heu après a definir ordres que dibuixaven quadrats, rectangles, circumferències i altres figures tancades. Cada figura necessita d'un procediment especial, que la dibuixa amb una mida de costat particular, com ara l'hexàgon en l'exemple següent.

```
procediment hexàgon :costat
  repeteix 6 [«
    avança :costat«
    gira.dreta 60«
  ]
fi
```

També heu vist que combinant figures senzilles era possible fer dibuixos força espectaculars, com el que el procediment següent realitza en base a hexàgon.

```
procediment super.hexàgon :costat
  repeteix 6 [«
    hexàgon :costat«
    gira.dreta 60«
  ]
fi
```

El mateix grau d'abstracció que representa dir que qualsevol figura plana tancada és un polígon pot assolir-se amb un procediment. A continuació, teniu el procediment poli, capaç de dibuixar una gran diversitat de polígons. Amb tan sols un procediment eliminem la raó de ser de gairebé tots els procediments que només sabien dibuixar una figura.

```
procediment poli :costat :angle
  avança :costat
  gira.dreta :angle
  poli :costat :angle
fi
```

Com podeu apreciar el procediment poli té una crida recursiva a la darrera línia, pel que mai s'aturarà, i farà que la tortuga dibuixi repetidament la mateixa figura. Prémer la tecla ESC és l'únic medi d'interrompre l'execució d'aquesta mena de procediments. Proveu les instruccions següents. Quina figura dibuixen? Experimenteu amb angles de diferents valors.

```
?poli 50 90
?poli 1 1
?poli 50 60
```

La recursivitat, tal com ha estat presentada fins ara, no és més que un mitjà d'implementar la repetició, contínua i

sense fi, d'un grup d'instruccions. A més de la recursió, i per construir estructures repetitives, el WIN-LOGO presenta l'ordre `mentre` juntament a l'ordre `repeteix`, que ja coneixeu. La sintaxi d'aquesta primitiva és certament inconsistent, exigint com a primer paràmetre una expressió lògica, però ficada dins d'una llista. Com a segon paràmetre ha de rebre una llista amb les instruccions que s'han d'executar mentre la condició expressada al primer paràmetre sigui veritat.

Proporcionant a l'ordre `mentre` la constant lògica `ver` com a primer paràmetre, podem provocar la repetició sense fi de les instruccions incloses al segon paràmetre. D'aquesta forma, l'anterior definició recursiva de `poli` pot transformar-se en un simple procés iteratiu.

```
procediment poli :costat :angle
  mentre ["ver] [«
    avança :costat«
    gira.dreta :angle«
  ]
fi
```

Anem ara a veure mitjans més civilitzats d'acabar l'execució d'un procediment, sense haver de recórrer a la tecla ESC. El procediment `poli.espiral`, amb què podreu dibuixar boniques espirals, té a la primera instrucció l'ordre `si`. Quan el costat de l'espiral sigui superior a una mida raonable, en aquest exemple 200, s'executarà l'ordre `acaba`. Aquesta primitiva fa que el procediment finalitzi per complet la seva execució, retornant el control al procediment que l'ha cridat, o al nivell superior si l'hem cridat nosaltres a la finestra de Treball.

```
procediment poli.espiral :costat :angle
  si :costat > 200 [acaba]
  avança :costat
  gira.dreta :angle
  poli.espiral :costat+5 :angle
fi
```

Proveu de dibuixar les espirals següents. Experimenteu amb angles de mides diferents.

```
?poli.espiral 1 90
?poli.espiral 1 120
```

Per tenir control sobre l'increment que ha de rebre el costat a cada volta, el procediment necessita d'un tercer paràmetre. Amb aquesta informació disponible, podeu provar de dibuixar espirals amb les línies més o menys atapeïdes.

```
procediment poli.espiral :costat :angle :i
  si :costat > 200 [acaba]
  avança :costat
  gira.dreta :angle
  poli.espiral :costat+:i :angle :i
fi
```

Tornant a la definició del procediment `poli`, anem a veure com parar la seva execució de forma menys dramàtica que amb la tecla ESC. La tècnica consisteix a mirar contínuament si una tecla ha estat premuda, i anar fent altres coses mentrestant. La funció `tecleig` retorna `ver` si l'usuari ha premut qualsevol tecla, i `fals` en cas contrari. La funció `caràcter.llegit` espera una tecla, i retorna el seu valor com una paraula d'una lletra. En executar el procediment següent, podreu observar que la línia amb l'ordre `escriu` s'executa correctament, cosa que no hagués passat cas d'haver premut la tecla ESC. En interrompre un procediment correctament, el control del

programa retorna al procediment que l'ha cridat, o al nivell superior només si nosaltres l'hem cridat directament dins la finestra de Treball.

```
procediment poli.stop :costat :angle
  mentre [no tecleig] [«
    avança :costat«
    gira.dreta :angle«
  ]
  (escriu [Has premut la tecla]«
          caràcter.llegit)
fi
```

Teorema del camí tancat

En fer servir el procediment `poli`, segur que heu observat que amb un angle de gir de $360/n$ sempre es dibuixa un polígon regular de n costats. Quan l'angle no és divisor de 360, aquesta relació no es manté. Per exemple, amb un angle de 144° es dibuixa una estrella de cinq puntes. Això ens permet observar que 5 x 114 =720°, i 720 és el doble de 360. D'aquest fet es pot deduir la fórmula **costats x angle = $\dot{360}$** —el nombre amb un punt damunt es llegeix "360 o múltiple de 360"—.

Si la tortuga, després de diversos girs i desplaçaments, torna a trobar-se a la mateixa posició i orientació inicial, podem estar segurs que el camí que ha seguit per fer el dibuix és tancat, i la figura dibuixada és una línia poligonal tancada. Això permet enunciar el conegut com teorema del camí tancat.

Teorema del camí tancat. El gir total realitzat al llarg de qualsevol camí tancat és un nombre enter múltiple de 360°.

I bé, us deveu preguntar, on anem a parar amb tot això? Serem capaços d'establir una condició d'acabament pel procediment `poli`. I ho farem tenint en compte el gir total, el nombre de graus acumulats que la tortuga ha girat. Quan aquest nombre sigui múltiple de 360, la figura estarà acabada.

```
posa.a "gir 0
procediment poli :costat :angle
  avança :costat
  gira.dreta :angle
  posa.a "gir :gir + :angle
  si (residu :gir 360) = 0 [acaba]
  poli :costat :angle
fi
```

Aquesta versió nova de `poli` fa servir una variable global per acumular el nombre total de graus girats. La condició de l'ordre `si` serà veritat quan la divisió del nombre de graus entre 360 tingui zero per residu. El procediment té una gran limitació, que és la d'obligar-nos a recordar la necessitat de posar a zero el contingut de `gir`, executant l'ordre següent.

```
?posa.a "gir 0
```

Tenim una solució per a aquest problema, i consisteix a fer servir una variable local en lloc d'una variable global. Per tal que una variable sigui local, el seu nom s'ha de passar com paràmetre a l'ordre `fes.local`.



Diem d'una variable que és *global* perquè el seu valor és accessible en qualsevol punt del programa. Una variable *local* només existeix dins d'un procediment, i tan sols és accesible des de dintre del mateix procediment.

La següent definició de `poli` s'encarrega de posar a zero la variable `gir`, i a més no té cap crida recursiva, fent

servir la primitiva `mentre` d'una forma especial. Fixeu-vos que la condició inicial sempre serà veritat, però que a la darrera línia es troba el test que permet finalitzar l'execució del procediment.

```
procediment poli :costat :angle
  fes.local "gir   ;declaració
  posa.a "gir 0    ;inicialització
  mentre ["ver] [«
    avança :costat«
    gira.dreta :angle«
    posa.a "gir :gir + :angle«
    si (residu :gir 360) = 0 [acaba]«
  ]
fi
```

Si acumular el valor del gir total, per determinar l'acabament del dibuix, és una estratègia basada en una propietat del camí, també és possible trobar una solució en base a les característiques de la figura dibuixada. Una qüestió present des de fa temps és determinar, a les figures dibuixades per `poli`, quina relació podem trobar entre el nombre de vèrtex i l'angle. Doncs bé, el nombre de vèrtex és igual al mínim comú múltiple de l'angle i 360, dividit per l'angle. La fórmula, amb expressió matemàtica, és ***vèrtex*** = MCM*(angle*,360)/*angle*. D'on surt aquesta fórmula? Bona pregunta. Intenteu respondre-la pel vostre compte. El cas és que ens permet saber el nombre de vèrtex abans de començar a dibuixar, i la podeu fer servir amb confiança encara que de moment no entengueu massa perquè funciona.

Per calcular el nombre de vèrtex necessitem calcular el mínim comú múltiple de dos nombres, i per solucionar-lo definirem una nova funció. El procediment `mcm`, per calcular el mínim comú múltiple de dos nombres, és un

bon exemple de com la força bruta també és útil. La seva estratègia no és gens refinada, però funciona. Observeu la forma en què controla la condició d'acabament del bucle.

```
procediment mcm :n :m
  (fes.local "i "múltiple "fi)
  posa.a "i 0
  posa.a "fi "fals
  mentre [no :fi] [«
    posa.a "i :i+1«
    posa.a "múltiple :i*:n«
    si (residu :múltiple :m) = 0«
       [posa.a "fi "ver]«
  ]
  retorna :múltiple
fi
```

Amb la funció mcm disponible, la definició final de `poli` queda de la forma següent. Fixeu-vos que la repetició s'obté a través de `repeteix`, i que per comoditat es fa servir una variable local.

```
procediment poli :costat :angle
  fes.local "r
  posa.a "r (mcm :angle 360) / :angle
  repeteix :r [«
    avança :costat«
    gira.dreta :angle«
  ]
fi
```

Fractals

Si la recursió és en el cas dels polígons una forma d'implementar la repetició, hi ha dibuixos on la recursió és un element estructural insubstituïble. Els fractals, per exemple, són dificils de dibuixar sense recórrer a la recursió, i amb aquesta són sumament fàcils de construir.

Un fractal és un model matemàtic o un objecte real, que manté la seva forma essencial, fragmentada i irregular, tot i variant l'escala d'observació. Els primers fractals van aparèixer a finals del segle XIX, però ha estat als anys setanta quan, a través dels treballs de Benoît Mandelbrot, han guanyat l'anomenada que tene avui en dia.

Els fractals també han cridat l'atenció dels artistes, i la música o l'animació gràfica per ordinador han trobat inspiració en aquests gràfics recursius.

Presentarem pocs exemples, senzills i elegants, que han esdevingut uns clàssics en el seu camp. Estudieu-los atentament, intentant relacionar les figures dibuixades i la forma en què aquestes es dibuixen, amb els procediments que les produeixen.

Tots els procediments determinen la condició d'acabament mitjançant una variable de nom `nivell`. Podem descriure els fractals com unes figures en què les seves parts poden considerar-se similars a la totalitat del dibuix, però a un nivell inferior. Amb el valor de la variable `nivell` determinem el nombre de parts autocontingudes que tindrà el dibuix. Així, la versió més simple d'un fractal serà la realitzada amb un `nivell` igual a 1. A mida que el valor de nivell s'incrementa, ho fa la complexitat del dibuix. Podeu intentar simular el resultat sobre d'un paper, començant amb pocs nivells de recursivitat.

La variable `mida` determina la grandària del dibuix. Haureu de provar els procediments amb diferents combinacions de valors per `mida` i `nivell`.

Els fractals ens porten a presentar els primer procediments amb múltiples crides recursives. Ja no ens trobem davant

de simples crides recursives per la cua, fet pel qual la complexitat dels procediments augmenta molt. Per poder seguir el fluxe d'execució en aquest context, us podeu ajudar de paper i llapis, o de les eines disponibles a l'entorn de Traçat. També pot ajudar el quadre de diàleg que obre la comanda **Arbre de Procediments...** del menú **Sistema**.

Corba *C*

Un dels fractals més coneguts és l'anomenada corba *C*. Una corba de nivell 0 es defineix com a igual a una recta, i una corba de nivell superior es defineix en funció altres dues corbes de nivell inferior. Fixeu-vos en què el valor de `nivell` es decrementat a cada nova crida recursiva, i que el procediment acaba al rebre un `nivell` de valor 0.

```
procediment corba.c :mida :nivell
  si :nivell = 0 [avança :mida acaba]
  corba.c :mida :nivell-1
  gira.dreta 90
  corba.c :mida :nivell-1
  gira.esquerra 90
fi
```

Proveu el procediment amb valors d'entrada diferents. A vegades, la relació entre `mida` i `nivell` no és la més òptima, i haureu d'anar fent proves. Les instruccions següents us mostren un fragment d'una sessió de treball amb el procediment `corba.c`. Endevineu d'on li ve el nom al fractal?

```
?inicia.dibuix corba.c 2 10
?inicia.dibuix corba.c 1.5 11
```

Floc de neu

Una corba autoreferent molt coneguda es la coneguda com a floc de neu. Es basa en un fractal que es dibuixa amb un procediment que té quatre crides recursives. No deixeu de fer el dibuix manualment, començant amb el nivell igual a 1, seguint amb nivell igual a 2, etc.

```
;;; Corba de tres dimensions
procediment fractal3 :mida :nivell
  si :nivell = 0 [avança :mida acaba]
  fractal3 :mida / 3 :nivell-1
  gira.esquerra 60
  fractal3 :mida / 3 :nivell-1
  gira.dreta 120
  fractal3 :mida / 3 :nivell-1
  gira.esquerra 60
  fractal3 :mida / 3 :nivell-1
fi
```

Fixeu-vos que a cada crida recursiva es decrementa el nivell, i la mida es divideix entre tres. Teclegeu les instruccions següents.

```
?inicia.dibuix fractal3 50 2
?inicia.dibuix fractal3 100 4
```

Amb simples repeticions de l'anterior fractal obtenim una interessant figura tancada. S'assembla força a un floc de neu.

```
procediment floc.de.neu :mida :nivell
    repeteix 3 [«
        fractal3 :mida :nivell«
        gira.dreta 120«
    ]
fi

?inicia.dibuix floc.de.neu 50 3
```

Corba del drac

La coneguda com corba del drac es dibuixa amb dos procediments recursius que es criden mútuament. El procediment `drac` enceta el dibuix amb una crida al procediment `drac.dreta`.

```
procediment drac :mida :nivell
    desapareix
    drac.dreta :mida :nivell
fi
```

Els dos procediments següents col·laboren per dibuixar la corba del drac. Son gairebé idèntics, fora de la instrucció de gir, en un cas a la dreta i en l'altre a l'esquerra.

```
procediment drac.dreta :mida :nivell
    si :nivell = 0 [avança :mida acaba]
    drac.esquerra :mida :nivell-1
    gira.dreta 90
    drac.dreta :mida :nivell-1
fi

procediment drac.esquerra :mida :nivell
    si :nivell = 0 [avança :mida acaba]
    drac.esquerra :mida :nivell-1
    gira.esquerra 90
    drac.dreta :mida :nivell-1
fi
```

Teclegeu les instruccions següents, i proveu diferents valors fins que aconseguiu fer el dibuix òptim pel vostre gust.

```
?inicia.dibuix drac 5 1
?inicia.dibuix drac 5 2
?inicia.dibuix drac 5 7
?inicia.dibuix drac 2 11
```

Coordenades cartesianes

El Logo també disposa d'un sistema de coordenades alternatiu a la geometria de la tortuga ja presentada. Aquesta nova metàfora de l'espai implica una tècnica de dibuix diferent a l'emprada fins ara. La geometria de la tortuga determina una concepció de l'espai relativa, mentre que la geometria cartesiana ofereix una concepció absoluta.

L'anomenada geometria cartesiana divideix el pla amb dues rectes perpendiculars, la intersecció de les quals passa a ser considerada el punt (0,0). El concepte de punt, definit com una parella de nombres, permet fer referència a qualsevol punt de l'espai de forma absoluta.

El punt on la tortuga es troba inicialment és el punt (0,0) i, en qualsevol moment, la primitiva `centre` mou la tortuga fins a aquest punt, des d'allà on es trobi. La primitiva `posa't` desplaça la tortuga fins a qualsevol punt arbitrari, representat amb una llista de dos nombres, i la funció `posició` informa del punt, també en forma de llista, on es troba la tortuga en un moment donat. Les funcions `coor.x` i `coor.y` informen del valor d'una coordenada aïllada, i les ordres `fes.x` i `fes.y` mouen a la tortuga horitzontal i verticalment. Observeu la següent definició del procediment `quadrat.cartesià`, basada en aquestes funcions. Compareu-la amb la definició basada en el gir de 90°.

```
procediment quadrat.cartesià :costat
  fes.y coor.y + :costat
  fes.x coor.x + :costat
  fes.y coor.y - :costat
  fes.x coor.x - :costat
fi
```

Dibuixar amb aquestes primitives implica una particular concepció de l'espai, diferent a la que representa guiar la tortuga amb moviments relatius a la seva posició. La combinació de les dues tècniques fa del Logo una eina especialment poderosa per l'exploració de l'espai.

Sumari

Taula 6-1. Primitives tractades

Nom	*Entrades*	*Efecte*
`acaba`	0	Dóna fi a una ordre
`caràcter.llegit`	0	Retorna un caràcter llegit del teclat
`centre`	0	Porta la tortuga al centre, punt (0,0)
`coor.x`	0	Retorna la coordenada x de la tortuga
`coor.y`	0	Retorna la coordenada y de la tortuga
`fes.local`	$1 \Rightarrow \infty$	Declara variables locals
`fes.x`	1	Mou la tortuga a una nova posició x
`fes.y`	1	Mou la tortuga a una nova posició y
`mentre`	2	Executa una llista d'instruccions mentre una condició sigui veritat
`posa't`	1	Mou la tortuga fins un punt donat
`posició`	0	Retorna el punt on és la tortuga
`tecleig`	0	Retorna `ver` si s'ha premut una tecla

El gir total realitzat al llarg de qualsevol camí tancat és un nombre enter múltiple de 360°.

Dibuixar fractals és molt fàcil amb procediments recursius.

El Logo també suporta una concepció cartesiana de l'espai.

Exercicis



Amb els mitjans presentats fins ara ja podeu plantejar-vos projectes molt interessants. Disposeu ja de moltes eines, útils per resoldre els problemes més diversos. A mida que les necessiteu les anireu assimilant.

Exercici 6-1. Modifiqueu el procediment `floc.de.neu` perquè pugui dibuixar flocs de neu amb un nombre variable de costats.

Exercici 6-2. Dibuixeu un rellotge, amb les seves corresponents minuteres. Feu que les agulles es moguin.

Exercici 6-3. Dibuixeu un rectangle tal que els vèrtex siguin els punts (-100,-50) i (100,50). Dibuixeu també el rombe que té per vèrtex els punts mitjos dels costats del rectangle.

Exercici 6-4. Utilitzant una variable de nom nivell, feu un procediment que dibuixi una sèrie de quadrats, l'un dintre l'altre.

Exercici 6-5. Feu una composició simètrica amb figures geomètriques, amb l'eix de simetria al centre de la pantalla.

Exercici 6-6. Recupereu i aprofiteu de nou el procediment `circumferència`, i feu un procediment per dibuixar rosetons.

Material complementari



Les possibilitats gràfiques del Logo van molt més enllà del que hem presentat en aquest capítol. A altres llibres sobre Logo trobareu moltes idees més per explorar l'espai. Els exemples següents en mostren algunes.

Exercici 6-7. Feu un procediment que dibuixi la gràfica de la funció y = x + 2. Proveu també amb la funció y = x * x + 10.

Exercici 6-8. El fitxer FRACTALS.LOG conté la definició dels fractals presentats en aquest capítol, i molts altres més. No deixeu d'estudiar-los tots.

Exercici 6-9. La fórmula *vèrtex* = MCM*(angle*,360)/*angle*, presentada anteriorment, no ha rebut la demostració pertinent. Llegiu de nou l'enunciat del teorema del camí tancat —clau per resoldre el problema—, i intenteu demostrar la fórmula en qüestió.

Exercici 6-10. Estudieu les funcions trigonomètriques primitives al Logo, com ara `sin` o `cos`. Aprofiteu-les per dibuixar diferents corbes sinusoidals.

Exercici 6-11. Milloreu el fractal que més us agradi afegint-li color. Com podeu relacionar els colors amb els fractals?

Exercici 6-12. Dibuixeu un paisatge simètric, amb procediments que dibuixin cap a la dreta o l'esquerra en funció d'un paràmetre. En cap cas dupliqueu els procediments de dibuix per obtenir la simetria.

7

PARAULES I LLISTES

El llenguatge Logo és dialecte del LISP, i va néixer a la mateixa universitat on el LISP ha gaudit de la màxima atenció, el prestigiós MIT. Aquest origen situa el Logo en la zona dels llenguatges aplicats en la intel·ligència artificial, i justifica la seva personalitat flexible i dinàmica. La pròpia paraula *logo* té una etimologia prou evident, i encara que els gràfics han suposat la glòria —o tragèdia— del Logo, el llenguatge va ser dissenyat pensant en el tractament del llenguatge, i no en els gràfics ni la tortuga.



En contraposició al tradicional concepte de càlcul aplicat als nombres, el Logo i el seu pare el LISP ofereixen el denominat *càlcul simbòlic*, aplicat a símbols en lloc de nombres. Així, mentre el càlcul numèric ens permet, per exemple, saber el valor d'una funció en determinat punt, el càlcul simbòlic fa possible obtenir la seva funció derivada.

Treballar amb símbols significa treballar amb paraules i llistes, matèria prima amb que el Logo ens permet representar el món, els objectes que el poblen i les relacions que entre ells mantenen. Encara que de forma informal, ja han estat presentades les paraules i les llistes. Heu fet servir les paraules per donar nom als

procediments, o per fer-les servir com a variables, i amb les llistes heu agrupat instruccions per passar-les com a paràmetre a primitives com `repeteix` o `mentre`.

Si considerem les paraules i llistes en elles mateixes, atenent als seus elements constitutius, podem considerar-les com un agregat ordenat d'elements més simples. Aquesta característica, comuna a paraules i llistes, ens permet considerar a tots els objectes Logo com sèries ordenades d'elements.



El diccionari defineix una *sèrie* com un seguit de coses que se succeeixen les unes a les altres relacionades d'alguna manera. Així les paraules són una sèrie de caràcters, i les llistes una sèrie d'objectes Logo —paraules, nombres o llistes—.

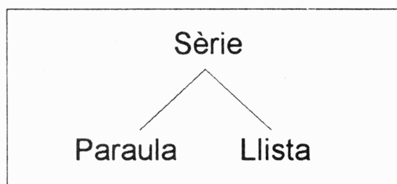


Figura 7-1. Sèries.

Per manipular paraules i llistes el Logo ofereix un simple però complet conjunt de funcions, que permeten accedir al contingut de les sèries, construir-ne de noves, o modificar el contingut de les existents. En general, les mateixes funcions que podem aplicar a les paraules funcionaran amb les llistes. De les funcions que es poden aplicar a més d'un tipus d'objecte en diem *funcions genèriques*. De totes formes, existeixen certes excepcions que més endavant coneixereu, i que es presenten sempre lligades a l'operació de construir una sèrie nova.

Paraules



Una *paraula* es defineix com una sèrie de caràcters, —lletres, números i signes de puntuació—, excloent-hi un conjunt petit de caràcters reservats. Els anomenats *caràcters especials* no poden formar part de les paraules, perquè serveixen per separar-les.

Per tal que el Logo consideri una paraula en ella mateixa, i no com el nom d'una primitiva o d'un procediment, l'hem de protegir amb les cometes.

```
?mostra "Hola!
 Hola!
```

Els diferents operadors aritmètics i relacionals consumeixen el major grup de caràcters especials. Aquesta representació dels operadors permet fer que "toquin" els seus operands, sense haver de posar un espai en blanc entremig. Com ja sabeu, no és possible definir nous operadors, com tampoc es poden definir nous caràcters especials. A la taula següent trobareu tots els caràcters especials que existeixen.

Taula 7-1. Caràcters especials

	espai	Separa les paraules.
;	punt i coma	Inicia els comentaris.
–	menys	Operador aritmètic de la resta.
$	dòlar	Protegeix la interpretació del caràcter següent com a especial. Això permet construir paraules amb caràcters especials com ara els espais.

( )	parèntesis	Permeten agrupar les expressions i alterar així l'ordre d'avaluació de les mateixes.
*	asterisc	Operador aritmétic de la multiplicació.
+	més	Operador aritmètic de la suma.
/	barra inclinada	Operador aritmètic de la divisió.
<	menor	Operador relacional 'menor que'.
=	igual	Operador relacional d'igualtat.
>	major	Operador relacional 'major que'.
[ ]	claudàtors	Delimiten l'inici i fi de les llistes.



El fet que els caràcters especials actuïn com a separadors entre paraules impedeix que puguin formar part d'una paraula. Existeix, però, un sistema per incloure els caràcters especials dins d'una paraula, consistent a precedir el caràcter en qüestió pel caràcter dòlar. Per incloure un caràcter de dòlar dins d'una paraula l'heu d'escriure dues vegades. Encara que la notació es torna una mica esotèrica, el sistema funciona, com podeu veure a l'exemple següent.

```
?mostra  "3$-2$ $=$ 1$$
 3-2 = 1$
```

Nombres

A tots els efectes, els nombres *són* paraules pel Logo. Això vol dir que totes les operacions que podem aplicar a

les paraules, són també aplicables als nombres. No succeeix el cas contrari: les operacions pròpies dels nombres, com ara la suma, no són aplicables a totes les paraules.

Aquesta assimilació dels nombres a les paraules pot causar algunes confusions. Observeu les instruccions següents. El que succeeix a la darrera instrucció, és una característica del Logo o un defecte del WIN-LOGO?

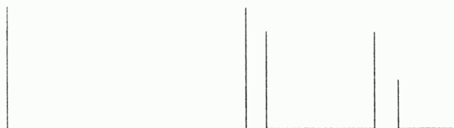
```
?mostra  és.nombre "12
 ver
?mostra és.paraula 12
  ver
?mostra "12 + 3
 15
?mostra "12 = 12
 fals
```

Llistes

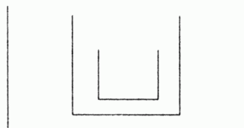
Una llista sempre comença per un claudàtor obert, i acaba amb un de tancat. Els elements continguts dins de les llistes poden ser paraules o altres llistes, pel que qualsevol objecte Logo pot ésser inclòs dins d'una llista.

Les llistes són prou flexibles per representar tota mena d'estructures de dades abstractes: taules, associacions, conjunts, arbres, etc. Potser aquesta flexibilitat pot produir una certa confusió inicial, però la gran llibertat que proporciona permet treballar de forma còmoda i expressiva.

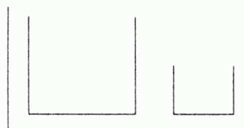
Les llistes dintre de llistes són com caixes de mides diferents unes dins les altres. Així, si considerem les tres caixes següents,



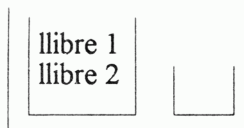
podem posar-les l'una dintre l'altre,



o posar les dues petites dins de la grosa però separades, amb el que tenim dues caixes buides dins d'una altra caixa.



Com és natural, podem posar llibres dins de la caixa mitjana, amb la qual cosa tindrem una caixa que té dins una caixa amb llibres i una caixa buida.



Amb el llenguatge Logo, l'últim exemple es pot representar amb la següent llista de dos elements: `[[llibre1 llibre2] []]`.

Manipulació de sèries

La gran homogeneïtat amb què el Logo tracta a paraules i llistes ens permetrà presentar les tècniques per

manipular-les de forma conjunta. En base a aquesta similitud, parlarem de *sèries* per referir-nos indistintament a paraules i llistes.

Estructura

Una sèrie està constituïda per un agregat ordenat d'elements diferents. És possible accedir directament al davant i al darrera de la sèrie, tant per veure l'objecte que s'hi troba com per posar-n'hi un de nou. Una sèrie pot crèixer, doncs, en dues direccions.

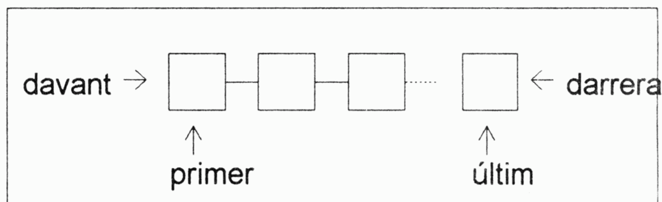


Figura 7-2. Estructura d'una sèrie.

Per accedir a un element arbitrari d'una sèrie cal, en principi, recórrer la sèrie començant per un dels seus extrems, fins arribar a l'element desitjat.

Accés

Les funcions bàsiques per accedir als elements d'una sèrie són `primer` i `últim`, que esperen una sèrie com a paràmetre, i retornen el seu primer o últim element. Si intentem accedir al primer o a l'últim element d'una sèrie buida, es produirà un error. Les prestacions d'aquestes funcions són limitades, però treballant en combinació poden ser de gran efectivitat. Tecle̊geu les instruccions següents, i fixeu-vos que en una sèrie de dos elements, la

funció `últim` permet accedir directament al segon element.

```
?mostra primer posició = coor.x
 ver
?mostra últim posició = coor.y
 ver
?mostra primer últim [Avui fa bon dia]
 d
```

El complement a `primer` i `últim` són les funcions `sense.primer` i `sense.últim`, que retornen el seu paràmetre sense el primer i a l'últim elements, respectivament. Ara podem definir, per exemple, una funció per accedir directament al segon element d'una sèrie.

```
procediment segon :sèrie
  retorna primer sense.primer :sèrie
fi
```

```
?mostra últim posició = segon posició
 ver
```

També és possible accedir a un element arbitrari d'una sèrie amb la funció `element`. Aquest accés, amb un índex numèric, és similar al que altres llenguatges practiquen per accedir a les seves estructures de dades, com ara els vectors o matrius. La funció `lloc` és el complement natural d'`element`, i retorna la posició d'un objecte dins d'una sèrie.

```
?mostra element 1 "Hola
 H
?mostra element 4 "Hola
 a
?mostra element 3 [la lletra H]
 H
?mostra lloc "H [Una H]
 2
```

La funció `element` provoca un error si li demanem que retorni un element situat a una posició inexistent dins de l'objecte. Si us agradaria més que en aquest cas la funció retornés una paraula o llista buida, en lloc de provocar un error, podeu fer servir la variant presentada a continuació. Com veureu, el nom del procediment comença per un punt. Aquesta convenció obeeix a la necessitat de diferenciar-lo del nom de la primitiva a la que emula. Poseu a prova la funció `.element` dins l'entorn de Traçat, i no deixeu d'observar el valor dels paràmetres que rep la funció en les diferents crides recursives.

```
procediment .element :n :sèrie
  si és.buida :sèrie [retorna :sèrie]
  si :n = 1 [retorna primer :sèrie]
  retorna .element :n-1 sense.primer :sèrie
fi
```

Recorregut

És una activitat molt corrent recórrer els elements d'una sèrie, i realitzar determinades operacions amb algun o la totalitat d'ells. Les tècniques a aplicar per recórrer la sèrie, ja sigui començant pel davant o el darrera, són similars.

Sumes

La seqüència d'instruccions necessàries per recórrer una sèrie d'esquerra a dreta, a fi de processar tots els seus elements, segueix l'esquema següent:

1. Si la sèrie és buida, acabar.
2. Processar el primer element de la sèrie.
3. Fer una crida recursiva, amb la sèrie sense el seu primer element.

Seguint aquesta estructura bàsica podem definir molts procediments interessants. Veieu, per exemple, la funció `suma.llista`, capaç de sumar tots els nombres continguts dins d'una llista. Fixeu-vos que, en rebre per paràmetre una llista buida, la funció retorna un zero, valor neutre de la suma. Quin valor hauria de retornar en aquest cas una funció similar a `suma.llista`, però que retornés la multiplicació dels nombres d'una llista?

```
procediment suma.llista :llista
  si és.buida :llista [retorna 0]
  retorna (primer :llista) +«
                  suma.llista sense.primer :llista
fi
```

Per provar la funció, crideu-la amb una llista de nombres per paràmetre.

```
?mostra suma.llista [1 2 3 4 5]
 15
```

Podem afegir a `suma.llista` la capacitat d'ignorar els elements de la llista que no siguin nombres. Fixeu-vos que ara, la primera crida recursiva del procediment ignora el primer element de la llista, mentre la segona l'inclou a la suma total.

```
procediment suma.llista :llista
  si és.buida :llista [retorna 0]
  si no és.nombre primer :llista [«
     retorna suma.llista sense.primer :llista«
  ]
  retorna (primer :llista) +«
                  suma.llista sense.primer :llista
fi
```

Amb aquesta nova qualitat, `suma.llista` pot ser útil per extreure la suma dels nombres situats dins d'una llista amb elements heterogenis.

```
?suma.llista [2 nois i 3 noies]
 5
?suma llista [2 més 2]
 4
```

Xifres

La seqüència d'instruccions necessàries per recórrer una sèrie de la dreta cap a l'esquerra, a fi de processar tots els seus elements, segueix l'esquema següent:

1. Si la sèrie és buida, acabar.
2. Processar l'últim element de la sèrie.
3. Fer una crida recursiva, amb la sèrie sense el seu últim element.

Vegeu un exemple que treu profit de la forma en què representem els nombres. El càlcul numèric ens ha acostumat a pensar en els nombres atenent, no a la seva representació, sinó en el valor que representen. El Logo ens permet mirar els nombres d'una altra forma. Per exemple, pot ser interessant disposar de funcions que, donada una xifra qualsevol, en retornin només un dels seus elements constitutius. Veieu les funcions següents, que retornen les unitats i desenes, respectivament, d'una xifra.

```
procediment unitats :xifra
  retorna últim :xifra
fi

procediment desenes :xifra
  retorna últim sense.últim :xifra
fi
```

Teclegeu els exemples següents:

```
?mostra unitats 321
```

```
 1
?mostra desenes 321
 2
```

Podem generalitzar les prestacions d'`unitats` i `desenes`, amb un procediment que permet extraure qualsevol element arbitrari d'una xifra, començant el compte pel nombre de més a la dreta.

```
procediment xifra :x :xifra
  si és.buida :xifra [retorna :xifra]
  si :x = 1 [retorna últim :xifra]
  retorna xifra :x-1 sense.últim :xifra
fi
```

Fixeu-vos que el procediment `xifra` treballarà correctament amb qualsevol base numèrica, binària, hexadecimal, etc. Encara més, `xifra` permet accedir a l'enèsim element de qualsevol sèrie —tant paraules com llistes—, començant el compte per la dreta.

Construcció

Les primitives que construeixen sèries noves són diferents per paraules i llistes. En el cas de les paraules existeix una sola funció, `paraula`, que accepta un nombre variable de paràmetres, i retorna una paraula confegida unint totes les paraules rebudes per paràmetre.

```
?mostra paraula "Ho "la
 Hola
?mostra (paraula "Bar "cel "ona)
 Barcelona
```

Pel que fa a les llistes, és possible afegir-hi elements nous tant pel devant com pel darrera, amb les funcions `anteposant` i `posposant`, respectivament.

```
?mostra anteposant "Fa [bon dia]
```

```
 [Fa bon dia]
?mostra posposant 4  [2 + 2 =]
 [2 + 2 = 4]
```

Un mitjà més general per confegir noves llistes l'ofereixen les funcions `llista` i `frase`. Totes dues accepten per paràmetre qualsevol nombre d'objectes a partir d'un mínim de dos, i retornen una llista amb tots els objectes dins. La diferència entre les dues funcions està en que `frase` "elimina" els claudàtors de les llistes que rep per paràmetre. Un exemple us ho aclarirà.

```
?mostra (llista [joan] "i [maria])
 [[joan] i [maria]]
?mostra (frase [joan] "i [maria])
 [joan i maria]
```

Les llistes són una estructura ideal per emmagatzemar col·leccions arbitràries d'objectes. Veiem, per exemple, com podríem mantenir una petita base de dades de compositors. Ho farem amb una variable global, que per contingut tindrà una llista amb els noms dels músics.

```
?posa.a "compositors [Bach Beethoven»
Brams]
?mostra :compositors
 [Bach Beethoven Brams]
```

Una vegada establerta la llista inicial de compositors, una operació habitual serà afegir un nou compositor.

```
?posa.a "compositors anteposant "Mozart»
:compositors
?mostra :compositors
 [Mozart Bach Beethoven Brams]
```

Per evitar el tedi que suposa actualitzar directament el contingut de `compositors`, podem definir un procediment que rebi per paràmetre el nom de compositor, i s'encarregui d'afegir-lo a la llista.

```
procediment nou.compositor :compositor
  posa.a "compositors«
    anteposant :compositor :compositors
fi
```

Ara és més fàcil afegir compositors a la llista.

```
?nou.compositor "Schumann
?mostra :compositors
 [Schumann Mozart Bach Beethoven Brams]
```

Ja disposem d'un sistema per representar les dades. Ara falta saber com extraure informació d'aquesta representació.

Consulta

El Logo ofereix diverses funcions per analitzar una sèrie qualsevol. Per exemple, la funció `núm.elements` permet esbrinar el nombre d'elements d'una sèrie, ja siguin les lletres d'una paraula, els dígits d'un nombre o el nombre d'elements d'una llista. Alerta doncs, en el cas de les llistes, perquè `núm.elements` compta tan sols el nombre d'elements que hi ha al nivell superior de la llista.

```
?mostra núm.elements [vocals [a e i o u]]
 2
```

Una primitiva molt important és la funció lògica `pertany`, que permet tractar les sèries com conjunts. `pertany` retorna `ver` si l'objecte que rep per primer paràmetre està inclòs en la sèrie que rep per segon paràmetre, i `fals` en cas contrari.

```
?mostra pertany "x [a e i o u]
 fals
```

Tornant a la llista de compositors, preguntes que podem necessitar respondre són saber el nombre total de músics

que hi ha, o esbrinar si un determinat artista és o no compositor.

```
?mostra núm.elements :compositors
 4
?mostra pertany "Picasso :compositors
 fals
```

Quan una determinada consulta és d'ús habitual, pot ser convenient definir una funció per encapsular les instruccions dins d'un procediment. D'aquesta forma, la consulta s'expressarà d'una forma abstracta, independentment de com estigui implementada la base de dades de compositors.

```
procediment és.compositor :compositor
  retorna pertany :compositor :compositors
fi
```

```
?és.compositor "Mozart
 ver
?és.compositor "Picasso
 fals
```

Vocals

Per extreure informacions més específiques d'una sèrie necessitem definir procediments especials. L'exemple que presentem compta el nombre de vocals que pugui haver dins d'una paraula. Fixeu-vos en l'ús que fa el procediment `conta.vocals` de la funció auxiliar `és.vocal`.

```
procediment és.vocal :lletra
  retorna pertany :lletra [ a e i o u à é è í ó ò ú ï ü ]
fi
```

```
procediment conta.vocals :paraula
  si és.buida :paraula [retorna 0]
  si és.vocal primer :paraula [«
     retorna 1 + conta.vocals sense.primer :paraula«
  ]
  retorna conta.vocals sense.primer :paraula
fi
```

Si acceptem que en una paraula només hi ha vocals i consonants, és fàcil esbrinar el nombre d'aquestes darreres lletres que pugui haver dins d'una paraula.

```
procediment conta.consonants :paraula
  retorna (núm.elements :paraula) - «
          conta.vocals :paraula
fi
```

Poseu en pràctica les noves funcions amb els exemples següents. Proveu també amb altres paraules.

```
?mostra conta.vocals "Hola
 2
?mostra conta.vocals "Barcelona
 4
?mostra conta.consonants "Barcelona
 5
```

Filtres

Seleccionar uns elements determinats d'una sèrie, i eliminar la resta, constitueix una típica operació de filtrat. Com a primer exemple de filtre, veieu la funció `si.és.nombre`. Espera una llista per paràmetre, i la retorna una vegada ha eliminat els objectes que no siguin nombres. De fet, el que fa `si.és.nombre` és construir una nova llista, començant pel final. Fixeu-vos que la primera instrucció del procediment retorna, si la condició d'acabament es compleix, una llista buida.

```
procediment si.és.nombre :llista
  si és.buida :llista [retorna []]
  si és.nombre primer :llista [
     retorna anteposant primer :llista«
                  si.és.nombre sense.primer :llista«
  ]
  retorna si.és.nombre sense.primer :llista
fi
```

El Logo disposa de la funció primitiva `elimina.element`, un simple filtre que elimina d'una llista la primera aparició d'un objecte qualsevol. Si el que voleu és eliminar totes les aparicions d'un objecte, i no tan sols la primera, feu servir la funció `elimina.tots`, que presentem a continuació.

```
procediment elimina.tots :objecte :llista
  si és.buida :llista [retorna [ ]]
  si primer :llista = :objecte [«
   retorna elimina.tots :objecte sense.primer :llista«
  ]
  retorna anteposant primer :llista«
     elimina.tots :objecte sense.primer :llista
fi
```



Fixeu-vos que l'estructura de tots els filtres és molt similar, amb variacions tan sols en el detall. També les funcions que recorren els diferents elements d'una sèrie tenen una estructura similar. Analitzeu els procediments "a distància", intentant veure el bosc darrera dels arbres. D'aquesta forma veureu com és molt més fàcil pensar en procediments nous per solucionar problemes nous.

Sumari

Les paraules i les llistes ofereixen possibilitats de treball similars a les dels llenguatges de la intel·ligència artificial.

Les paraules i les llistes estan constituïdes per una sèrie d'elements enllaçats.

Taula 7-2. Primitives tractades

Nom	*Entrades*	*Efecte*
`anteposant`	2	Afegeix un objecte al davant d'una llista
`element`	2	Retorna l'objecte enèsim d'una sèrie
`elimina.element`	2	Elimina la primera aparició d'un objecte dins d'una llista
`frase`	$2 \Rightarrow \infty$	Confegeix una llista nova
`llista`	$2 \Rightarrow \infty$	Confegeix una llista nova
`lloc`	1	Retorna la posició d'un objecte dins d'una llista
`núm.elements`	1	Compta el nombre d'elements d'una sèrie
`paraula`	$2 \Rightarrow \infty$	Confegeix una nova paraula
`pertany`	2	Determina si un objecte forma part d'una sèrie
`posposant`	2	Afegeix un objecte al darrera d'una llista
`primer`	1	Retorna el primer objecte d'una sèrie
`sense.primer`	1	Treu el primer objecte d'una sèrie
`sense.últim`	1	Treu l'últim objecte d'una sèrie
`últim`	1	Retorna l'últim objecte d'una sèrie

Exercicis



Encara que vàlids per ells mateixos, heu de veure els exercicis següents com una pràctica necessària i d'ús general. Penseu que moltes primitives esperen llistes per paràmetre, i també hi ha moltes funcions que retornen llistes. El que ara aprengueu ho podreu aplicar en situacions molt diferents.

Exercici 7-1. Feu un procediment que escrigui 100 vegades la frase *No parlaré a classe*. Feu que el text s'escrigui a la impressora.

Exercici 7-2. Definiu una funció que retorni el tercer element d'una sèrie.

Exercici 7-3. Feu una funció que, donada una paraula, retorni una llista amb les lletres de la paraula. Feu una funció que faci l'operació inversa.

Exercici 7-4. Feu un procediment que, donada una llista de nombres, retorni el més petit de tots els nombres.

Exercici 7-5. Feu un procediment que, donada una llista amb noms de nois i una altra de noies, faci parelles a l'atzar.

Exercici 7-6. Feu un programa traductor, que donada una paraula catalana la tradueixi a un idioma diferent.

Material complementari



El material que hem presentat sobre manipulació de paraules i llistes és simple però força complet. Les possibilitats de treball que ofereix el Logo en aquest camp són molt grans, i si desitgeu avançar en l'estudi del Logo en aquesta direcció, trobareu llibres molt interessants a la bibliografia especialitzada.

Exercici 7-7. El procediment per la formació del plurals que presentem a continuació és completament innocent, i no contempla la veritable complexitat que presenta l'idioma català en aquest camp. De totes formes, funciona en la majoria dels casos.

```
procediment plural :paraula
  si últim :paraula = "a [«
   ↳retorna paraula sense.últim :paraula«
                   "es«
  ]
  retorna paraula :paraula "s
fi
```

Milloreu el procediment `plural`, tant com pugueu, fins que accepti les múltiples excepcions que pot tenir la formació dels plurals amb substantius i adjectius.

Exercici 7-8. Feu un procediment per convertir un nombre qualsevol a la notació dels romans, i un altre que faci la tasca inversa.

Exercici 7-9. Definiu una funció que accepti un paràmetre i el retorni completament del revés. Li podeu posar per nom `capgira`. Hauria de treballar d'una forma similar a la següent.

```
?mostra capgira "Adéu
 uédA
```

```
?mostra capgira [Maria i Josep]
 [Josep i Maria]
```

Un consell: definiu dues funcions, una per capgirar paraules i una altra per capgirar llistes, de nom `capgira.paraula` i `capgira.llista`. Crideu aquestes funcions des de `capgira`, en funció del tipus de sèrie que rebi per paràmetre.

Exercici 7-10. Feu un procediment que tradueixi un nombre a la seva escriptura catalana, amb guionets inclosos.

Exercici 7-11. Les funcions `primer` i `últim` provoquen un error si reben com a paràmetre un objecte buit. Definiu una versió personal d'aquestes funcions, que retornin l'objecte buit en lloc de provocar un error. Fixeu-vos que aquesta versió serà coherent amb les propietats dels conjunts que diuen que el conjunt buit és subconjunt de qualsevol conjunt, i que qualsevol conjunt és subconjunt de si mateix.

Exercici 7-12. Definiu una funció, que donada una llista de nombres, retorni una llista amb la suma i la mitjana dels seus valors.

8

EXPERIÈNCIES



Quin és l'estat de salut del Logo? Quines activitats es duen a terme actualment amb el llenguatge? Aquest capítol estarà dedicat a presentar experiències diferents, amb l'afany d'intentar respondre les preguntes anteriors i altres de similars.

La visió del Logo que es pugui derivar d'allò que presentarem serà forçosament parcial i com podeu imaginar, s'estan realitzant moltes altres activitats i experiències importants amb el Logo a més de les que apareixeran en aquestes pàgines.

Pla Logo

Amb el nom de "Pla experimental d'Introducció de la Informàtica a l'EGB a través del lenguatge Logo" es va realitzar a Catalunya el projecte més ambicios d'inserció del Logo al currículum escolar. Pel seu gran abast, tant territorial com temporal, i el rigor amb què es van estudiar els resultats del treball, aquesta experiència és un referent obligat a l'hora de parlar de l'ús del Logo a Catalunya.

Per conèixer els detalls complets de com es va desenvolupar aquesta experiència podeu llegir l'article següent:

BENEDITO, V.; TORRE, S. de la (1990) *Seguimiento del Plan Experimental de Introducción de la Informática en la EGB a través del lenguaje Logo*. Infodidac, núm. 10, pp.35-42.

Les conclusions de l'experiència, i altres reflexions complementàries, les trobareu a l'article següent:

BENEDITO, V.; CEA, F.; TORRE, S. de la (1991) *El potencial cognitivo del lenguaje Logo*. Infodidac, núm. 16, pp.20-26.

Control

Una àrea diferent a la dels gràfics típics del Logo, i que ha rebut força atenció, ha estat la de la tecnologia de control. Mitjançant petites instal·lacions realitzades amb interruptors, commutadors, motors, engranatges, etc., i que són controlades des de l'ordinador a través d'una interfície especial, és possible construir micromóns molt variats. Guiar un cotxe a través d'un circuit, o controlar els moviments d'una sínia, són algunes de les activitats que caracteritzen l'ús de la tecnologia de control amb el Logo.

Des del seu neixement l'any 1986, el Programa d'Informàtica Educativa (PIE) va començar una estreta col·laboració amb el "Pla experimental d'Introducció de la Informàtica a l'EGB" i amb les escoles implicades en l'experiència. Des d'aleshores el PIE ha dut a terme diverses actuacions relacionades amb el Logo, i que ara no mencionarem exhaustivament. La tecnologia de

control ha rebut una atenció especial dintre d'aquestes actuacions, pel que farem esment de l'estat actual d'aquest tema.

Al llarg del curs 1992-93 el PIE està realitzant, sota el nom genèric d'*EL TALLER*, unes activitats d'introducció de les tecnologies de control per nens i nenes dels quatre als dotze anys. A través de deu micromons diferents es treballen, entre d'altres temes, les posicions absolutes, jocs d'estratègia, la conducció d'un cotxe, etc., i es presenten activitats concretes per 1er, 2on i 3er cicle de primària.

Pel que fa a l'ensenyament secundari, també el PIE ha dut a terme activitats de control amb Logo. Com a resultat d'aquestes experiències està disponible, per a qui pugui interesar, el conjunt de primitives que fan possible el control de diferents dispositius electromecànics, adaptades especialment al WIN-LOGO.

Geometria

Amb l'objectiu d'oferir materials de treball acabats, aplicables directament a la classe, el Programa d'Informàtica Educativa ha publicat un quadern de fitxes titulat *Geometria al Cicle mitjà,* dissenyat com a reforç a la classe de geometria, i en cap cas com un substitut de la mateixa. La referència completa d'aquest quadern és la següent:

COLLADO, R.; GARCIA, R.; MIRALDA, S.; MORAGAS, M.; MORALES, M.; PALAU, R.; DE PEDRO, F.; SAURA, I. *Geometria al Cicle mitjà.* (1992) Barcelona: Programa d'Informàtica Educativa.

Aquesta publicació pretén convertir l'ordinador en un laboratori de geometria, on s'experimenti i posi en pràctica la teoria, de forma semblant a com treballen, per exemple, les ciències naturals.

Després d'una introducció, en què es presenten els trets metodològics que s'han de tenir presents per treure el màxim profit al quadern, es presenten vint-i-una fitxes de treball, totes amb una estructura similar, i que de forma progressiva presenten els conceptes geomètrics fonamentals. El nivell de coneixement del llenguatge Logo necessari per a la realització de les fitxes és realment elemental, pel que el llenguatge no suposa un dificultat afegida a les que pugui presentar la geometria en ella mateixa.

Està en preparació un nou quadern de fitxes adreçat al cicle superior, continuació natural del quadern ja editat, i especialment adaptat a la nova normativa derivada de la reforma educativa.

La tauleta

Al primer capítol ja hem parlat de la tauleta sensible i de les seves aplicacions relacionades amb el Logo, i aquí no ho tornarem a fer. Tan sols us volem recordar la disponibilitat d'aquesta eina, especialment adaptada als més petits, i que fa possible l'accés als recursos informàtics als infants no lectors i a altres persones amb necessitats educatives especials.

Logo a l'escola

Un estimulant escrit apareix en les actes de la Conferència sobre Tecnologia de la Informació en l'Educació,

celebrada a Barcelona el novembre de 1992. Una de les comunicacions explica com l'escola AULA de Barcelona ha desenvolupat al llarg dels últims deu anys una completa programació d'activitats amb el llenguatge Logo, i arriba a cobrir tot l'espectre d'edats. Començant amb nens de cin anys, i arribant a 8è d'EGB, tots els alumnes de l'escola AULA fan servir el Logo. Aquesta activitat, segons la pròpia comunicació, ha fet possible que els alumnes traslladin l'actitud heurística fomentada a l'aula d'ordinadors a la resta de matèries, i sobre tot, que aquest esperit de descoberta sigui un tret característic del seu pensament i de la seva personalitat.

La referència completa de la comunicació és la següent: FORTUNY, M.; ROCA, C. *LOGO A L'ESCOLA: Reflexions sobre una experiència estimulant i continuada durant 10 anys.* (1992) Barcelona: European conference about information technology in education: a critical insight. Proceedings, vol. 1.

LOGO EXCHANGE

Amb aquest nom —*LOGO EXCHANGE*— es publica una revista quatrimestral sota els auspicis de la International Society for Technology in Education (ISTE). La publicació es dirigeix als educadors que fan servir el Logo en la seva pràctica diària, i recull articles que descriuen experiències realitzades a tot el món.

Si podeu llegir amb anglès no dubteu de fullejar aquesta publicació, el millor sistema de saber l'actualitat mundial del Logo. Per realitzar una subscripció a *LOGO EXCHANGE* us podeu dirigir a l'ISTE, escrivint a l'adreça següent:

ISTE
1787 Agate Street
Eugene, OR 97403-1923
USA

Música

Durant el curs 1991-92, l'autor d'aquestes mateixes pàgines va disposar d'una llicència per estudis amb la finalitat d'implementar una extensió musical al Logo. El resultat d'aquest treball és ***SONOR***, una extensió al Logo que fa possible executar música amb moderns sintetitzadors, de forma simultània a les altres tasques que el Logo pot realitzar, com ara dibuixar. La informació necessària per l'execució musical pot agafar-se d'un fitxer MIDI —un estàndard per la representació d'informació musical en fitxers—, o ser definida a través d'un petit conjunt de noves primitives.

La relació del Logo amb la música és tan antiga com el propi llenguatge, però el preu dels instruments musicals digitals ha estat un problema durant molts anys. Avui en dia això ja no es així, i més d'un centenar d'escoles de Catalunya han estat dotades d'aquests equipaments pel Programa d'Informàtica Educativa.

Si teniu interès en incorporar música als vostres treballs amb el Logo, o explorar les possibilitats del llenguatge en l'educació musical, ***SONOR*** és l'eina que ho permet fer. No dubteu d'adreçar-vos directament al seu autor per tenir accés a una còpia del treball.

Cloenda



Ara que ja s'acaba aquesta *Invitació al Logo* és hora de fer balanç. La visió del Logo que podeu tenir en aquest moment ha de ser amb tota seguretat força diferent a la del principi. Ara ja podeu decidir si el Logo us interessa o no, i si us sembla una eina didàctica vàlida i digna de ser posada en pràctica. Discutiu les estratègies que us semblin més adequades per posar en pràctica el Logo en el vostre nivell educatiu.

En aquestes pàgines hem intentat oferir reflexions i consells sobre la pràctica educativa amb el Logo, però la major part del text ha versat sobre el llenguatge Logo i les seves característiques com a tal. Si us decidiu a tirar endavant, la principal mancança que podeu necessitar cobrir està relacionada amb la teoria educativa associada al Logo. Respecte a aquest tema, si bé podeu trobar material excel·lent a la bibliografia especialitzada, serà la vostra pràctica i reflexió la que us orientarà millor.

Pel que fa a la continuació de l'estudi del llenguatge Logo, la bibliografia us pot dirigir cap a les lectures adequades. A més, per començar teniu a la vostra disposició el proper capítol, que podeu considerar com una gran col·lecció d'exercicis avançats. Atenció: no cal que us enfronteu a aquest capítol si no teniu seguretat amb el llenguatge i una atracció clara per la programació.

9

ALGORITMES



Encara que aquesta *Invitació al Logo* té per vocació un caràcter divulgador, hem inclòs tot un capítol, aquest que ara comença, de caire avançat. Per limitacions d'espai, les properes explicacions no tindran tot el detall que la complexitat dels exemples potser demanaria tenint en compte el vostre nivell actual, i per això us demanem que accepteu disculpes anticipades.

Podeu veure aquest capítol com un avanç, una mena de propaganda del que pot fer-se amb el Logo. També podeu considerar els exemples del capítol com una col·lecció d'eines, que podeu fer servir segons els vostres interessos i necessitats. Per fer servir aquestes eines no us cal entendre del tot com funcionen internament. Tan sols us cal saber el seu nom, les entrades que volen els procediments, i quins valors retornen les funcions. Res més.

Algorísmia

Entenem per algoritme una sèrie de d'instruccions per realitzar una tasca concreta a través d'un procés mecànic i finit. La paraula algoritme es deriva del nom del matemàtic àrab Al-Jwãrizmî, o més exactament, Abû Abd

Allãh Muhammad ibn Mûsa Al-Jwãrizmî. En veure aquest nom, segur que heu entes per que la paraula va evolucionar...

En parlar d'algorísmia és inevitable comentar el caire algorísmic de les receptes de cuina. Inclouen unes instruccions que, seguides al peu de la lletra (?), permeten cuinar un plat determinat. Cuinar, igual que moltes altres activitats humanes, necessita seguir procediments de treball precisos i predefinits.

Els exemples d'aquest capítol han estat triats pel fet de fer servir algoritmes interessants. Uns tenen interès per ser tradicionals, històrics; altres per ser molt generals i aprofitables en situacions diverses; altres per resoldre de forma sorprenent problemes molt complexos, etc.

Tot el material del capítol es troba al disquet que acompanya aquest manual, i abans de començar cada un dels apartats trobareu el nom del fitxer on són els procediments. Podeu estalviar molta feina recuperant directament aquest fitxers, però no deixeu de treballar algun dels exemples dins l'editor.

Nombres

La primera sèrie d'exemples mostra diferents eines de caire aritmètic. Els nombres sencers són una font inesgotable d'especulació intel·lectual, i sembla mentida la poca falta que fan, la major part de les vegades, els nombres reals.

Divisors d'un nombre

Fitxer: DIVISORS.LOG

Començarem per mostrar un algoritme basat en la força bruta. Els ordinadors estan acostumats a realitzar el que els demanem sense queixar-se, per avorrit que sigui. Per trobar els divisors d'un nombre, l'ordinador només ha d'anar provant repetidament si un nombre divideix a un altre nombre, i això ho sap mirant si el residu que queda en dividir els dos nombres és zero. L'única mostra d'intel·ligència de l'algoritme és no intentar trobar divisors més grans que la meitat del nombre. En efecte, el divisor més gran que podem trobar d'un nombre és la seva meitat, i això si el nombre és parell.

El procediment `divisors` s'ajuda d'un procediment auxiliar, que és el que fa la feina dura. Aquesta és una tècnica molt habitual en el Logo, i la trobareu repetidament al llarg d'aquest capítol. El procediment `divisors` retorna una llista amb tots els divisors d'un nombre, per la qual cosa és una eina que podeu integrar tal com està on faci falta.

```
;;; Càlcul de tots els divisor d'un nombre
procediment divisors :n
  retorna div.aux :n quocient :n 2
fi

;;; Retorna una llista amb els divisors
procediment div.aux :n :m
  si :m < 2 [retorna [1]]
  si (residu :n :m) = 0 [«
     retorna anteposant :m«
                               div.aux :n :m-1«
  ]
  retorna div.aux :n :m-1
fi
```

De forma oportunista, podem fer servir la funció `divisors` per saber si un nombre és primer. Si un nombre no té divisors (tan sols l'u), és que és un nombre primer.

```
procediment és.primer :n
      retorna divisors :n = [1]
fi
```

La funció `és primer` retornarà `ver` o `fals`, segons si el seu paràmetre és un nombre primer o no. No és la forma més eficient de decidir si un nombre és primer, però funciona i no ha donat cap feina de preparar, ja que aprofita una funció que ja existia. Si estudieu l'algoritme que fa servir la funció `divisors`, veureu que també serviria per determinar si un nombre és primer. Apart de canviar el nom dels dos procediments pels d'`és.primer` i `és.primer.aux`, tan sols s'hauria de modificar la primera i tercera línia de la funció auxiliar. La primera canviaria per

```
si :m < 2 [retorna "ver]
```

i la tercera per

```
[retorna "fals]
```

D'aquesta forma la funció retornaria un resultat lògic en lloc d'una llista. Fàcil, no?

Màxim comú divisor

Fitxer: MCD.LOG

Parlant d'algoritmes no podia faltar l'algoritme d'Euclides pel càlcul del màxim comú divisor. Expressat informalment, l'algoritme és el següent:

1. Dividir el nombre més gran pel més petit.

2. Si el residu de la divisió és zero, el nombre més petit és el màxim comú divisor.
3. Si el residu no és zero, agafeu el més petit dels nombres i el residu de la divisió. Torneu a repetir el procés amb aquests dos nombres.

Provem d'expressar l'algoritme amb el llenguatge Logo?

```
;;; Màxim comú divisor segons
;;; l'algoritme d'Euclides
procediment mcd :n :m
    si :m = 0 [retorna :n]
    retorna mcd :m residu :n :m
fi
```

La funció mcd es pot fer servir a qualsevol expressió on calgui saber el màxim comú divisor de dos nombres. No deixeu de provar-la des de la finestra de Treball.

Factorial

Fitxer: FACT.LOG

Un altre algoritme recursiu tradicional és el que calcula el factorial d'un nombre. El factorial d'un nombre és igual a 1 si el nombre és 1 o 0. En els altres casos, el factorial és igual al nombre multiplicat pel factorial del nombre menys 1. La formula matemàtica següent ho explica millor que les paraules.

$$f(n) = \begin{cases} 1, & \text{per } n = 0 \\ 1, & \text{per } n = 1 \\ n \times f(n-1), & \text{per } n > 1 \end{cases}$$

Figura 9-1. Definició matemàtica de la funció factorial.

El procediment que calcula el factorial és pràcticament una translació directa de la fórmula matemàtica al llenguatge Logo.

```
;;; Factorial d'un nombre
procediment fact :n
    si :n < 2 [retorna 1]
    retorna :n * fact :n-1
fi
```

Proveu aquesta funció amb nombres grans i petits. Quin és el nombre més gran del qual el WIN-LOGO pot calcular el factorial?

Mostreig aleatori

Fitxer: MOSTREIG.LOG

La simulació per ordinador és un camp molt prometedor, amb grans possibilitats educatives. Les diverses manifestacions de l'atzar tenen un paper fonamental en qualsevol simulació, i el procediment que aquí presentem permet simular la tria aleatòria de diversos elements d'un conjunt, sense que es produeixin repeticions.

La funció `mostreig` retorna una llista de nombres, amb tants elements com la seva primera entrada indiqui. Els

nombres estaran entre zero i el valor indicat a la segona entrada. Així, per exemple, l'expressió `mostreig 6 48` retornarà una llista amb sis nombres entre 0 i 48.

```
;;; Simulació d'un mostreig aleatori
procediment mostreig :mostres :màxim
  si :mostres = 0 [retorna []]
  (fes.local "seleccions "pròxim)
  posa.a "seleccions mostreig :mostres-1 :màxim-1
  posa.a "pròxim atzar :màxim
  si pertany :pròxim :seleccions [«
     posa.a "pròxim :màxim«
  ]
  retorna anteposant :pròxim :seleccions
fi
```

Podem fer servir la funció `mostreig` com a base per simular el joc de la primitiva. Tal com `mostreig` està ja funciona, si acceptem que tots els nombres estaran disminuïts en un. Juguem una butlleta sencera?

```
?connecta "prn
?repeteix 6 [escriu mostreig 6 48]
?desconnecta
```

Amb una còpia sobre paper és mes fàcil omplir la butlleta. Si us toca, segur que estimareu el Logo!

Sèrie de Fibonacci

Fitxer: FIB.LOG

Un problema presentat pel matemàtic medieval Leonard de Pisa, més conegut com Fibonacci, al seu llibre *Liber abaci*, va inspirar a molts matemàtics al llarg de la història. El problema és el següent:

«Quantes parelles de conills tindrem al final d'un cert any, si, començant amb una parella, cada parella produeix

cada mes una parella que pot reproduir-se després de dos mesos d'existència?»

La solució del problema va donar lloc a la coneguda com "sèrie de Fibonacci" 1, 1, 2, 3, 5, 8, 13, ..., U_n..., en la que $U_n = U_{n-1} + U_{n-2}$, és a dir, cada un dels termes de la successió, començant pel 2, és igual a la suma dels dos termes que el precedeixen. La fórmula següent defineix aquesta sèrie de forma precisa.

$$f(n)=\begin{cases}1, & \text{per } n = 0\\ 1, & \text{per } n = 1\\ f(n-1)+f(n-2), & \text{per } n > 1\end{cases}$$

Figura 9-2. Definició matemàtica de la funció de Fibonacci.

Començarem per definir la funció de la forma més similar a la definició matemàtica, fet que implica fer servir doble recursió.

```
;;; Funció de Fibonacci amb doble recursió
procediment fib :n
  retorna si :n < 2 [1][(fib :n-1) + (fib :n-2)]
fi
```

L'eficàcia d'aquesta implementació és petitísima, ja que la multiplicació de crides recursives és realment immensa. És fàcil modificar la funció per convertir la doble recursió en simple recursió de cua amb l'ajuda d'un procediment auxiliar.

```
;;; Funció de Fibonacci amb recursió de cua
procediment fib.cua :n
    si :n < 2 [retorna 1]
    retorna fib.cua.aux 1 1 1 :n
fi

procediment fib.cua.aux :fo :fn :i :n
    si :i = :n [retorna :fn]
    retorna fib.cua.aux :fn :fo+:fn :i+1 :n
fi
```

La relació entre la funció de Fibonacci i la proporció àuria va atraure no tal sols matemàtics, sinó arquitectes, escultors i músics. La proporció daurada $(1+\sqrt{5})/2$ i el seu invers $(1-\sqrt{5})/2$ va permetre a Binet presentar la següent definició de la funció de Fibonacci.

$$f(n)=\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1}-\left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right]$$

Figura 9-3. Funció de Fibonacci segons la fórmula de Binet.

A continuació, teniu el procediment que calcula la funció de Fibonacci segons la fórmula de Binet. Abans de copiar-lo, seria interessant que vosaltres intentéssiu definir-lo com a exercici.

```
;;; Funció de Fibonacci segons la formula de Binet
procediment fib.binet :n
  fes.local "a5
  posa.a "a5 arrel 5    ;per eficiència
  retorna part.entera (/ (- potència (/ (1+:a5) 2)«
                                    :n+1 «
                            potència (/ (1-:a5) 2)«
                                    :n+1)«
                         :a5)
fi
```

Pot resultar molt instructiu comparar la velocitat d'execució de les tres variants de la funció de Fibonacci presentades. Una d'elles és escandalosament lenta. Quina? Que la fa tan lenta? Feu servir el següent procediment per realitzar les proves (trobareu aquest procediment al fitxer TEST.LOG).

```
procediment test :expr
  (fes.local "inici "retorn "total)
  posa.a "test "acció
  posa.a "inici temps
  recull "error [posa.a "retorn executa :expr«
               posa.a "test "funció]
  posa.a "total temps - :inici
  escriu frase [Temps d'execució:] :total
  si :test = "funció [escriu.seguit [Valor retornat:]«
                     mostra :retorn]
fi
```

Un cop definida la funció `test`, les instruccions següents mesuraran els temps d'execució, al vostre ordinador, de les diverses implementacions presentades de la funció de Fibonacci.

```
?test [fib 7]
?test [fib.cua 7]
?test [fib.binet 7]
```

La funció `test` apareix aquí únicament en qualitat d'eina. L'única pretensió que tenim és que us sigui útil i la feu servir, però si us ve de gust estudieu el seu comportament. Podeu aprendre com definir procediments que es comportin com a ordres o com a funcions alhora, tal com fan les primitives `si` o `executa`.

Arrel entera

Fitxer: ARREL.LOG

Segons el diccionari, l'aritmètica és l'estudi dels nombres naturals i de les operacions d'addició, substracció, multiplicació, divisió entera, potenciació i extracció d'arrels enteres entre aquests nombres. Si repasseu les operacions enumerades a la definició, descobrireu que el Logo desconeix la forma de calcular arrels senceres. Li ensenyem?

```
;;; Arrel entera d'un nombre
procediment arrel.entera :n
  arrel.entera.aux 1 0 :n-1
fi

procediment arrel.aux :p :k :n
  si :n < 0 [retorna :k]
  retorna arrel.aux :p+2 :k+1 :n-(:p+2)
fi
```

Proveu la funció i veureu que funciona realment. Encara que no és difícil dir *com* funciona l'algoritme que fa servir la funció, costa de veritat entendre *perquè* funciona. Fixeu-vos que tan sols fa servir sumes i restes per calcular el seu valor. Us veieu amb cor de trobar una descripció breu i entenedora de l'algoritme amb què treballa la funció `arrel.entera`? Si ho feu, modifiqueu els noms

de les variables emprades als procediments a fi de fer més explícita la seva tasca.

No cal que us amoïneu si no enteneu massa bé —o gens— l'algoritme que permet a la funció `arrel.entera` calcular els seu resultats. Simplement, feu servir la funció. Potser, més endavant, necessitareu calcular l'arrel entera d'un nombre. Aleshores, només haureu de recuperar el fitxer ARREL.LOG i recordar el nom i la sintaxi de la funció. A qui preocuparà aleshores com aquesta fa la seva feina?

Paraules i llistes

Encara que hem començat parlant de nombres, el llenguatge Logo escau potser millor a problemes de caire simbòlic. El Logo és un dialecte del LISP, llenguatge principal de la intel·ligència artificial, i aquesta paternitat s'ha de notar d'alguna forma.

Desordre

Fitxer: REMENA.LOG

Simular l'acció de barrejar una baralla de cartes, o qualsevol acció similar, consistent a desordenar un conjunt d'elements, pot ser la base de programes de simulació interessants. La funció `remena` retorna una llista amb els mateixos elements que tingui la llista que rebi com a paràmetre, però completament desordenats.

```
;;; Desordena una llista
procediment remena :llista
  retorna remena.aux :llista []
fi
```

```
procediment remena.aux :llista :retorn
  fes.local "tria
  si és.buida :llista [retorna :retorn]
  posa.a "tria un.al.atzar :llista
  retorna anteposant :tria«
                     elimina.element :tria :llista
fi

;;; retorna un element a l'atzar
procediment un.al.atzar :llista
  retorna element 1+(atzar núm elements :llista)«
                  :llista
fi
```

Conjunts

Fitxer: CONJUNTS.LOG

Les llistes s'adapten molt bé per representar diferents conjunts. Les operacions fonamentals amb conjunts són de molt fàcil implementació amb el Logo; a continuació presentem les més importants. Les funcions següents esperen dues entrades, els conjunts sobre els quals operar. Retornen un conjunt nou, una llista, resultat d'aplicar l'operació sobre les entrades del procediment.

```
;;; Retorna el conjunt unió
procediment unió :A :B
  si és.buida :A [retorna :B]
  si pertany primer :A :B«
    [retorna unió sense.primer :A :B]
  retorna anteposant primer :A«
                     unió sense.primer :A :B
fi
```

```
;;; Retorna el conjunt intersecció
procediment intersecció :A :B
  si és.buida :A [retorna []]
  si pertany primer :A :B [«
     retorna anteposant primer :A«
                      intersecció sense.primer :A :B«
  ]
  retorna intersecció sense.primer :A :B
fi

;;; Retorna el conjunt diferència
procediment diferència :A :B
  si és.buida :A [retorna []]
  si pertany primer :A :B«
     [retorna diferència sense.primer :A :B]
  retorna anteposant primer :A«
                    diferència sense primer :A :B
fi
```

El cas de la funció `és.subconjunt` és diferent al de les tres funcions anteriors, ja que retorna un valor lògic, és a dir, les paraules `ver` o `fals`.

```
;;; Determina si un conjunt és subconjunt d'altre
procediment és.subconjunt :A :B
  si és.buida :A [retorna "ver]
  retorna veres.totes pertany primer :A :B«
                     és.subconjunt sense.primer :A :B
fi
```

Dos conjunts són iguals si tenen els mateixos elements, independentment de l'ordre en què aquests es trobin dins la llista. Per determinar si dos conjunts són iguals podem aprofitar la propietat dels conjunts que diu que un conjunt sempre es té a si mateix com a subconjunt.

```
;;; Determina si dos conjunts són iguals
procediment mateix.subconjunt :A :B
    retorna (veres.totes és.subconjunt :A :B
                         és.subconjunt :B :A)
fi
```

Si un conjunt es caracteritza per no tenir cap element repetit, pot ser que en alguna ocasió ens faci falta convertit una simple col·lecció d'elements, amb possibles repeticions, en un conjunt. La funció `treu.repetits` fa aquesta feina per nosaltres.

```
;;; Elimina els element repetits d'una llista,
;;; convertint-la en un conjunt
procediment treu.repetits :col.lecció
  si és.buida :col.lecció [retorna []]
  si pertany primer :col.lecció«
             sense.primer :col.lecció«
     [retorna treu.repetits sense.primer :col.lecció]
  retorna anteposant primer :col.lecció«
                treu.repetits sense.primer :col.lecció
fi
```

Reconeixement de patrons

Fitxer: `SIMILAR.LOG`

Com ja sabeu, el Logo disposa de l'operador d'igualtat per comparar dos objectes i determinar si són iguals. A vegades, però, cal més flexibilitat a l'hora de fer les comparacions. La funció `són.similars` compara dues llistes, i accepta dins la primera dos elements especials. Un interrogant és equivalent a qualsevol element, i l'admiració equival a un o més elements. Veiem alguns exemples de com podem fer servir aquesta funció.

```
?mostra són.similars [? bon ?] [Fa bon dia]
 ver
?mostra són.similars [? bon ?] [És bon nen?]
 ver
?mostra són.similars [? bon ?] [Quin bon dia que fa!]
 fals
?mostra són.similars [? bon !] [Quin bon dia que fa!]
 ver
```

La funció conté quatre comprovacions, realitzades per la primitiva `si`. Estudieu cada una d'aquestes comprovacions per separat. Les dues primeres no poden ser modificades, però la tercera i quarta donen tractament a l'interrogant i a l'admiració respectivament, i podrien ser eliminades si a la vegada prescindíssim dels elements especials què donen suport. Si totes les comprovacions fracassen, la funció retorna `fals`.

```
;;; Reconeixement de patrons
procediment són.similars :patró :frase
  ;; Si les dues llistes són buides són iguals
  si veres.totes és.buida :patró«
                 és.buida :frase«
    [retorna "ver]
  ;; Si una llista no és buida i l'altre si ho és
  ;; no són similars
  si vera.alguna és.buida :patró«
                 és.buida :frase«
    [retorna "fals]
  ;; Si les primeres paraules són iguals
  ;; continuem amb la resta
  si vera.alguna (primer :patró = "?)«
                 (primer :patró = primer :frase)«
     [retorna són.similars sense.primer :patró«
                           sense.primer :frase]
  ;; El més difícil: doble recursió per
  ;; acceptar el patró múltiple (!)
  si primer :patró = "! [«
    retorna vera.alguna són.similars«
                           sense.primer :patró«
                           sense.primer :frase«
                        són.similars«
                           :patró«
                           sense.primer :frase«
  ]
  ;; Arribats aquí el patró i la frase no són similars
  retorna "fals
fi
```

Permutacions

L'algoritme que proporciona totes les permutacions d'una llista qualsevol d'elements està darrera de la solució de molts problemes que necessiten explorar sistemàticament totes les solucions possibles, com ara decidir quina figura cal moure en el joc d'escacs.

Presentem dues implementacions diferents d'aquest algoritme. La primera és la més simple, i la que us aconsellem que estudieu si us interessa el tema. La segona presenta una interessant funció aplicativa, i treu profit de les variables d'abast dinàmic, una prestació obscura del llenguatge Logo.

Primera versió

Fitxer: PERM1.LOG

El procediment `permuta` no fa res més que cridar el procediment auxiliar `permuta.aux`, que és qui fa la feina de veritat. Aquest procediment es crida recursivament de forma repetida. Si us sembla adequat agafeu un paper i un llapis i dibuixeu un arbre de les crides que `permuta.aux` fa. Entendreu millor el procediment si ho feu així.

```
;;; Permutacions d'una llista
;;; Versió amb recursió i iteració combinades
procediment permuta :llista
    permuta.aux :llista []
fi
```

```
procediment permuta.aux :llista :retorn
    (fes.local "x "elements "element)
    posa.a "elements :llista
    mentre [no és.buida :elements] [
        posa.a "element primer :elements
        posa.a "x elimina.element :element :llista
        si és.buida :x«
          [mostra anteposant :element :retorn]«
          [permuta.aux :x anteposant :element :retorn]
        posa.a "elements sense.primer :elements
    ]
fi
```

Observeu com el procediment `permuta.aux` mostra la tècnica que s'ha de fer servir per recórrer tots els elements d'una llista sense fer servir la recursió. L'estructura general per recórrer la llista és la següent:

```
mentre [no és.buida :llista] [«
  ... «
  posa.a :llista sense.primer :llista«
]
```

Fixeu-vos que en sortir d'aquest bucle el contingut de la variable `llista` és la llista buida.

Segona versió

Fitxer: PERM2.LOG

Si hem inclòs una segona versió de l'algoritme per obtenir totes les permutacions dels elements d'una llista és per presentar un procediment molt interessant, `aplica`, i per fer un petit pecat. Aquest pecat no és més que el de fer servir variables amb abast dinàmic. Més endavant explicarem que són les variables amb abast dinàmic. Primer parlarem de les funcions aplicatives.

Una situació freqüent en treballar amb el Logo és la d'haver de repetir una acció determinada amb cada un dels elements d'una llista. Imagineu que disposeu d'un procediment de nom `aplica`, que ha de rebre com a primera entrada el nom d'una ordre, i com a segona una llista d'elements sobre els quals aplicar l'ordre (que ha d'esperar una entrada). Un exemple us aclarirà aquesta situació.

```
?aplica "escriu [a e i o u]
 a
 e
 i
 o
 u
```

Un mecanisme de repetició com el proporcionat per `aplica` pot facilitar moltes de les tasques habituals dins dels programes de Logo. Amb una ordre com `camina`, presentada a continuació, que espera com a entrada una llista amb un nombre de passes a avançar i un nombre de graus a girar; podria ser molt fàcil fer dibuixos complets emmagatzemats dins d'una variable. Com a mostra presentem una nova forma de dibuixar un quadrat... Serà l'última?

```
procediment camina :passes.i.gir
  avança primer :passes.i.gir
  gira.dreta últim :passes.i.gir
fi
```

```
?posa.a "quadrat [[50 90] [50 90] [50 90] [50 90]]
?aplica "camina :quadrat
```

Veiem finalment la definició del procediment `aplica`. La seva clau es troba en la primitiva `executa`, que com el seu nom indica executa un llista d'instruccions. Fixeu-vos

que la primitiva `executa` ofereix una via d'accés directa a l'intèrpret de llenguatge Logo.

```
;;; Utilitat de caire general
procediment aplica :.ordre :.llista
    si és.buida :.llista [acaba]
    executa llista :.ordre primer :.llista
    aplica :.ordre sense.primer :.llista
fi
```

No us estranyi el punt amb què comença el nom dels paràmetres del procediment. Són necessaris per evitar conflictes, en la mida del possible, amb els noms de variables que poguessin aparèixer dins la llista d'instruccions que s'ha d'executar.

Anem finalment a veure la segona versió de l'algoritme de permutació. L'aspecte més polèmic d'aquesta implementació es troba en l'ús que fa de les variables d'abast dinàmic. Què vol dir això? Vol dir que les variables locals del procediment `permuta.1`, definit més endavant, són accessibles des de dins del procediment `permuta.aux`. Això no és cap trampa, i és un recurs legítim, encara que controvertit, del llenguatge Logo. Aquest ús de les variables no és només perillós, sinó que fa molt més complicada la compressió dels programes. L'altre complicació que trobareu està en el fet que `permuta.1` crida a `permuta.aux`, encara que indirectament a través d'`aplica`, i que `permuta.aux` crida a `permuta.1`.

Per què tanta complicació, deveu pensar, si la primera implementació de l'algoritme feia la mateixa feina de forma molt més clara? Dons bé, tothom ha de tenir dret a cometre un pecat de tant en tant, o no?

```
;;; Permutacions d'una llista
;;; Versió amb recursió creuada i
;;; variables d'abast dinàmic
procediment permuta :llista
    permuta.1 :llista []
fi

procediment permuta.1 :elements :retorn
    aplica "permuta.aux :elements
fi

procediment permuta.aux :element
    fes.local "x
    posa.a "x elimina.element :element :elements
    si és.buida :x«
        [mostra anteposant :element :retorn]«
        [permuta.1 :x anteposant :element :retorn]
fi
```

Sumari

Taula 9-1. Primitives tractades

Nom	*Entrades*	*Efecte*
`recull "error`	2	Permet recuperar errors d'execució
`mentre`	2	Implementa l'execució iterativa, amb una condició inicial
`executa`	1	Executa una llista d'instruccions
`temps`	0	Retorna el temps d'execució del WIN-LOGO
`escriu.seguit`	$1 \Rightarrow \infty$	Escriu les seves entrades sense començar una nova línia

En ser el Logo un llenguatge recursiu, permet implementar directament i amb gran facilitat tots els algoritmes que es defineixin de forma recursiva.

El Logo és un llenguatge poderós i expressiu. És un dialecte del llenguatge LISP, i ha heretat d'aquest la capacitat d'enfrontar-se a problemes de caire simbòlic.

Les noves primitives introduïdes al llarg d'aquest capítol no han rebut l'atenció que tal vegada necessitaven. No deixeu de consultar la seva definició al *Manual de Referència* del WIN-LOGO, o a l'ajuda del propi programa.

Exercicis



Entendre tan sols una part del material presentat en aquest capítol pot ser ja un exercici més que suficient per vosaltres. Per aquesta raó, abans d'intentar fer els exercicis (que no seran fàcils), treballeu a fons el contingut del capítol, seguint les recomanacions següents a l'hora d'estudiar els procediments definits.

- Teclegeu tots els exemples que us interessin dels presentats al llarg del capítol, i experimenteu amb els procediments fent petits programes de prova.
- Organitzeu acuradament el material nou en fitxers diferents, afegint al codi els comentaris que considereu adequats.
- Intenteu implementar procediments similars als presentats, o variacions o extensions dels mateixos.
- Afegiu un control més gran sobre els paràmetres que poden rebre els procediments. Per exemple, què passa si la funció `fibonacci` rep un paràmetre negatiu?

Exercici 9-1. Coneixent la relació següent, definiu un procediment per calcular el mínim comú múltiple de dos

nombres. Aprofiteu la funció `mcd` definida en aquest capítol.

$$\text{MCM}(p,q) \times \text{MCD}(p,q) = p \times q$$

Exercici 9-2. Modifiqueu la primera implementació de la funció de Fibonacci presentada perquè acumuli, en una variable global de nom `crides`, el nombre de crides recursives que produeix. Amb aquesta modificació realitzada, les instruccions següents haurien d'informar-vos del nombre de crides necessàries per cacular l'expressió `fib 7`.

```
?posa.a "crides 0
?mostra fib 7
?mostra :crides
```

Exercici 9-3. Modifiqueu la funció `permuta` per tal que, en lloc d'escriure les permutacions dels elements d'una llista, acumuli els seu resultat dins d'una variable global, que posteriorment podreu consultar.

Exercici 9-4. Modifiqueu el procediment `aplica` perque, en el cas que el seu primer paràmetre sigui una funció i no una ordre, retorni una llista amb tots els resultats retornats per la funció.

Exercici 9-5. Feu un programa que dibuixi totes les figures tancades amb un nombre de costats que sigui divisor sencer de 360. Aprofiteu la funció `divisors`, i feu que el programa s'aturi, esperant una tecla, i esborri el contingut de la finestra de Gràfics, abans de fer cada dibuix.

Exercici 9-6. Feu un programa per resoldre equacions de segon grau.

Material complementari



El material complementari d'un capítol de caire avançat com és aquest, ha de ser necessàriament ambiciós. Considereu els exercicis següents més com a projectes, que poden demanar força temps per ser desenvolupats, que com a exercicis que podeu resoldre al llarg d'una setmana.

Alguns dels projectes proposats poden necessitar molts més coneixements i informació que la que aquesta *Invitació al Logo* us hagi pogut proporcionar. Per això, si decidiu treballar-los podeu haver d'acudir a altres llibres o estudis complementaris. Bona sort!

Exercici 9-7. Feu un programa per jugar a les cartes.

Exercici 9-8. El WIN-LOGO suporta tortugues múltiples i pot detectar col·lisions entre elles. Implementeu, amb diverses tortugues, una simulació d'una gàbia amb animals diferents, uns més violents i altres menys, i determineu la probabilitat que dos animals es barallin si es troben en passejar. Quan de temps pot sobreviure determinada població d'animals amb les condicions inicials que vosaltres establiu?

Exercici 9-9. Segur que recordeu de l'escola el procediment per calcular nombres primers conegut com el *garbell d'Eratòstenes* —si no és així, busqueu la seva descripció en qualsevol llibre elemental d'aritmètica_. Fent servir l'algoritme d'Eratòstenes feu un programa que calculi tots els nombres primers menors de deu mil.

Exercici 9-10. EL WIN-LOGO disposa de primitives per guiar la tortuga a l'espai tridimensional. Definiu

procediments per dibuixar figures geomètriques tridimensionals, com ara cubs, piràmides, etc.

Exercici 9-11. Feu un programa per generar poesies de forma aleatoria.

Exercici 9-12. Feu un dibuix en moviment d'un coet espacial.

BIBLIOGRAFIA

En aquest apartat no pretenem oferir una completa bibliografia sobre el llenguatge Logo i les seves aplicacions. Trobareu informació d'aquesta mena a alguns dels llibres comentats. El propòsit d'aquest capítol és presentar alguns textos que considerem recomanables. No són necessàriament llibres de gran actualitat, però el seu interès s'ha mantingut inalterat al llarg del temps.

No existeixen llibres de Logo adaptats especialment al WIN-LOGO. Els exemples que podeu trobar als llibres presentats estaran pensats per altres implementacions del llenguatge, i fins i tot en altres idiomes. Res d'això us ha d'espantar. Al contrari, veure estils diferents és educatiu i necessari per tenir una visió global del que suposa el Logo en les seves diferents implementacions.

PAPERT, SEYMOUR

Desafío a la mente

Computadoras i educación

Buenos Aires: Ediciones Galápago (1981)

Aquest llibre és el culpable de tot. Editat originalment al 1980, va suposar un revulsiu considerable respecte a les pràctiques d'ensenyament assistit per ordinador aleshores en ús, i va presentar el Logo com a la solució revolucionària i definitiva per facilitar l'ús dels ordinadors als nens i nenes i per a l'aprenentatge de les matemàtiques.

S'ha acusat Papert de "passar-se" en les promeses sobre allò que el Logo havia de fer possible. Fins hi ha qui a

tractat aquest llibre de "pamflet". Papert propugnava, per exemple, que el Logo podia afavorir o accelerar el desenvolupament cognitiu dels nens, i això no s'ha pogut demostrar. Si be és cert que sense Papert el Logo no hagués existit, potser també ha estat el responsable de que hom estigui, actualment, una mica desencisat amb el Logo.

Desafío a la mente és de lectura obligada per conèixer el punt de partida de l'aventura que l'hi ha tocat viure al Logo. L'aventura continua, però tot va començar amb aquest llibre.

REGGINI, HORACIO C.

Alas para la mente

LOGO: Un lenguaje de computadoras y un estilo de pensar

Buenos Aires: Ediciones Galápago (1982)

Encara que l'impacte del Logo a l'Estat espanyol no és menyspreable, a altres països el Logo ha donat lloc a fenòmens especials. L'Argentina s'ha distingit, després dels EUA, per la vitalitat amb que ha mantingut obert l'interes pel Logo. La figura principal del moviment Logo a aquest païs ha estat Horacio Reggini í el llibre que presentem, una completa introducció al llenguatge i a les seves aplicacions. *Alas para la mente* va ser el primer llibre important sobre Logo escrit originalment en llengua castellana, raó per la qual la seva influència a tot l'Estat espanyol ha estat molt important.

RODRIGUEZ ROSELLÒ, L.

LOGO, de la tortuga a la inteligencia artificial

Madrid: Vector Ediciones (1986)

Aquest voluminós llibre representa segurament el major i més variat compendi d'exemples del llenguatge Logo publicat a l'Estat espanyol. El seu principal defecte és fer servir una versió del Logo en llengua anglesa. De totes formes, aquest factor afavoreix el grau d'universalitat dels exemples, que poden ser adaptats sense esforç a qualsevol versió del Logo, com ara el WIN-LOGO. En aquest llibre hi trobareu desenvolupats alguns temes que en aquesta *Invitació al Logo* tan sols han estat esbossats, i d'altres que ni tan sols s'han mencionat. La bibliografia, encara que antiga, és força completa.

ABELSON, H.; DISESA, A.

Geometria de la tortuga

El ordenador como medio de exploración de las Matemáticas

Madrid: Ediciones Anaya Multimedia (1986)

Aquest llibre va ser publicat originalment per l'Institut de Tecnologia de Massachusetts, l'any 1980. Realitza una profunda investigació de la geometria de la tortuga, demostrant les possibilitats exploratòries del Logo en el camp de les matemàtiques. S'adreça a nivells de batxillerat o superiors, fet que limita certament el seu abast. Amb aquest llibre, la imatge despectiva del Logo com una eina vàlida tan sols per nens, deixa de tenir raó d'existir.

GROS, B.

Aprender mediante el ordenador

Barcelona: Biblioteca Universitària de Pedagogia. PPU (1987)

Aquest llibre conté una presentació molt ampla de la teoria educativa lligada al Logo, fet pel qual és un complement obligat a altres llibres més orientats a presentar simplement el llenguatge de programació. També hi trobareu una descripció del paper que el Logo va tenir com a revulsiu respecte a l'anomenat EAO o ensenyament assistit per ordinador. Respecte a aquest tema, el llibre distingeix entre aprendre *de* l'ordinador, i aprendre *amb* l'ordinador. Aquest darrer aprenentatge seria el que el Logo fa possible, i es distingiria per plantejar un ús realment innovador dels ordinadors.

NOTES



Generalitat de Catalunya
Departament d'Ensenyament
Programa d'Informàtica Educativa

Ref: **C8-A1**